Comparison of Hologram Calculation Implementations for Wavefront Recording Plane Method Using Look-up Table Method and Direct Calculation Method

Hidenari Yanagihara, Tomoyoshi Shimobaba, Takashi Kakue, Tomoyoshi Ito

Graduate School of Engineering, Chiba University, 1-33, Yayoi-cho, Inage-ku, Chiba, 263-8522, Japan
Keywords: electro-holography, computer-generated hologram, wavefront recording plane, look-up table

ABSTRACT

We evaluated calculation times of computer-generated holograms based on wavefront recording plane method using several implementations in the combination of look-up table method and direct calculation method in order to realize real-time electro-holography system. We confirmed that there are different characteristics between CPU and GPU implementations.

1 INTRODUCTION

Three-dimensional (3D) displays [1,2] are promising techniques for realizing a surgical-support system in the medical field and next-generation 3D television systems. Among 3D displays, we focus on electro-holographic displays [3-5] which can reconstruct moving pictures by displaying holograms on a spatial light modulator (SLM). Holograms displayed on the SLM, which is referred as to computer-generated holograms (CGHs), can be generated by a numerical simulation of light on a computer. A direct calculation method for generating point-cloud-based CGH superposes the complex amplitude of light emitted from 3D object points using integral equations.

Realizing a real-time holographic 3D display is necessary to accelerate CGH calculation. Many methods have been reported to accelerate CGH calculation, such as dedicated hardware for point-cloud-based CGH calculation [6], look-up table (LUT) method [7], and wavefront recording plane (WRP) [8] method. In the CGH calculation by the dedicated hardware in Ref. [6], a CGH is calculated by running many dedicated processors in parallel. In the LUT method of Ref. [7], we pre-calculate complex amplitudes emitted from single object points with different axial distances and store them in the LUT. Subsequently, these complex amplitudes are superposed by reading from the LUT according to the coordinate information of the 3D objects.

In CGH calculations by using the LUT method and the direct calculation method, it has not been detailed whether the two methods had different characteristics. In this paper, we aim to compare the CGH calculations by using the LUT method and direct calculation method. In addition, we apply these implementations to the WRP method and evaluate them.

2 METHOD

2.1 Wavefront recording plane (WRP)

In electro-holography, information about 3D object is recorded as CGHs by calculating the propagation and interference of light on a computer. In our study, we considered 3D objects as point cloud and used the WRP method to accelerate CGH calculation. Figure 1 shows the schematic diagram of the WRP method. In the first step, we place a WRP close to the 3D objects and record the complex amplitude of the 3D objects in the WRP. The complex amplitude formed on the WRP, $u_w(x_w, y_w, z_w)$, can be expressed by

$$u_w(x_w, y_w, z_w) = \sum_{j=1}^{N} \exp\left[i \frac{2\pi}{\lambda} \frac{(x_j - x_w)^2 + (y_j - y_w)^2}{2d_j} \right], \quad (1)$$

where (x_w, y_w, z_w) denote the coordinates on the WRP, (x_j, y_j, z_j) denote the coordinates of the *j*-th point-light source of the 3D objects, $d_j = z_j - z_w$ denotes the perpendicular distance between the WRP and *j*-th point-light source, N denotes the number of point clouds, i denotes the imaginary unit, and λ denotes the wavelength of light wave. In the second step, we calculate the light propagation from the WRP to the CGH using the Fresnel diffraction.

The radius of the spreading region for the *j*-th point-light source on the WRP, W_j, can be expressed as follows:

$$W_j = |d_j| \tan(\theta_0), \quad (2)$$

where $\theta_0 = \sin^{-1}(\lambda / (2p))$ denotes the maximum diffraction angle for reconstructing the 3D objects and p denotes the pixel pitch of the CGH. We need to judge whether the object light passes through the circular region with the radius W_j during the CGH calculation or not. In our study, we set the square region inscribed in the circular region shown in Fig. 1(b) to mitigate the judgment of the circular region. The side length of the square region, D_j, can be expressed as

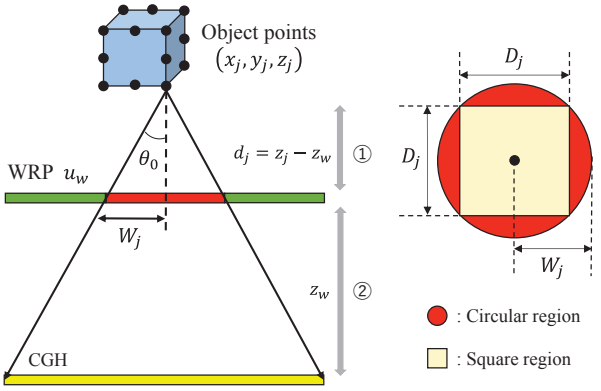
$$D_j = \frac{2W_j}{\sqrt{2}}. \quad (3)$$

The average of D_j for all the 3D object points, $\overline{D}$, can be

expressed as

$$\bar{D} = \frac{1}{N}\sum_{j=1}^{N} D_j. \qquad (4)$$

In our study, we performed CGH calculation using the square region shown in Fig. 1(b).

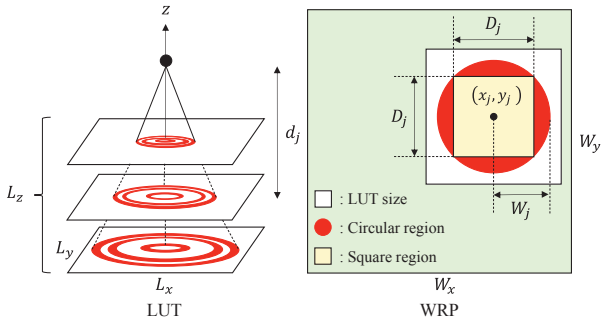


(a) Outline of the WRP method. (b) Calculation region.

Fig. 1: Schematic diagram of the WRP method.

2.2 Look-up table (LUT)

In the LUT method, we store the complex amplitudes, which are pre-calculated by using Eq. (1) and d_j, into the LUT. Figure 2 shows the schematic diagram of the LUT method. W_x and W_y denote the number of WRP pixels, respectively. When the LUT size is $L_x \times L_y$ pixels and the number of the depth is L_z, the amount of memory for the LUT requires $L_x L_y L_z$. We accumulate the complex amplitude while reading from the LUT. As shown in Fig. 2(b), we determine the calculation region in the WRP by using x_j, y_j, d_j, and D_j which are parameters of the 3D objects.



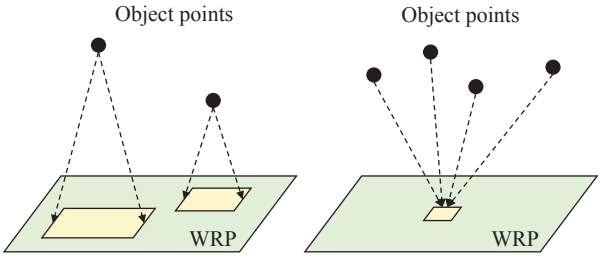
(a) Outline of the LUT method. (b) Calculation region.

Fig. 2: Schematic diagram of the LUT method.

2.3 Accumulation of the complex amplitude

As shown in Fig. 3, there are two methods for accumulating complex amplitudes: "Scatter" and "Gather". In "Scatter" as shown in Fig. 3(a), we accumulate a complex amplitude on the WRP emitted from a single object point, and then we repeat the same procedure for next object points. We determine the calculation region of the complex amplitude in the WRP by using x_j, y_j, d_j, and D_j which are parameters of the 3D objects. Because the calculation region is different for each object point, the amount of memory writing requires $N\bar{D}^2$. This accumulation can be implemented by both direct calculation method of Eq. (1) and LUT method.

In "Gather" as shown in Fig. 3(b), we calculate a WRP pixel by accumulating the light wave emitted from all the object points, and we repeat this procedure for all the WRP pixels. In this calculation, we need to determine which object points pass through the WRP pixels. Because the complex amplitudes are accumulated for each WRP pixel, the amount of memory writing requires $W_x W_y$. This accumulation can only be implemented by direct calculation method of Eq. (1).



(a) Scatter. (b) Gather.

Fig. 3: Schematic diagram of accumulating the complex amplitude.

3 EXPERIMENT

In this study, we compared the direct calculation method of Eq. (1) with the LUT method. We evaluated the calculation times by changing the number of object points N. We used Microsoft Windows 10 Enterprise as an operating system, a CPU (Intel Core i7-6700 with 3.4 GHz), and a GPU (GeForce GTX 1080). We used Microsoft Visual Studio Enterprise 2015 [9] and computer unified device architecture (CUDA) [10] as integrated development environments for the PC and GPU, respectively.

We set the resolution of the WRP to $W_x = W_y = 4{,}096$, the resolution of LUT to $L_x = L_y = 256$, and the depth of LUT to $L_z = 256$. We set the distance between the WRP and the CGH to $z_w = 300$ mm, the depth range of the 3D objects to $3\ \text{mm} \le d_j \le 30\ \text{mm}$, the pixel pitch to $p = 8\ \mu\text{m}$, and the wavelength of light to $\lambda = 532$ nm. Table 1 shows the symbols of the combination of calculation methods and hardware.

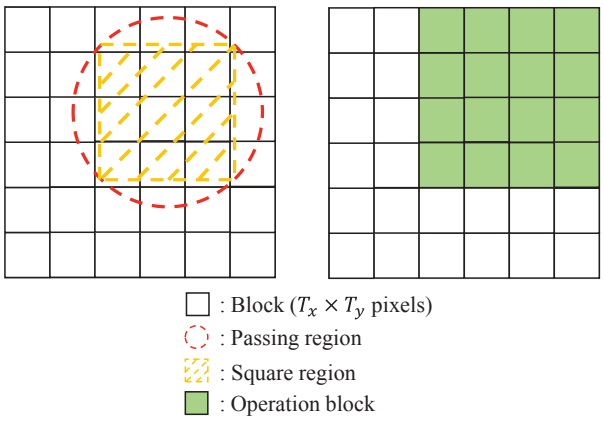
Next, we explain the accumulation of the complex amplitude in GPU implementation. For parallel processing using CUDA, we need to set the number of threads that form a "block" and the number of blocks that form a "grid". In our study, we parallelized the pixels of WRP. Figure 4 shows the schematic diagram of the accumulation in GPU implementation. T_x and T_y denote the number of pixels (threads) in a block, and B_x and B_y denote the number of blocks in a grid. In our study, we set the number of threads to $T_x = T_y = 32$. If light wave emitted from an object point is distributed as the red dashed circle in Fig. 4(a), we approximate the circle

region as the yellow square region. Subsequently, the square region is divided into blocks as shown in Fig. 4(b). The GPU calculates the complex amplitude in each block in parallel.

The number of blocks is different between the “Scatter” and “Gather” implementations. In the “Scatter” implementation, the passing region of light as shown in Fig. 4(a) was formed within the LUT. Therefore, the number of blocks was $B_x = L_x / T_x = 8$ and $B_y = L_y / T_y = 8$. The “Gather” implementation must determine whether a complex amplitude calculation in a WRP pixel is required. Therefore, the number of blocks was $B_x = W_x / T_x = 128$ and $B_y = W_y / T_y = 128$.

Table 1: Symbols of the combination of calculation methods and hardware.

	Symbol	Combination
CPU	C1	Direct calculation method of Eq. (1) by using “Scatter” with a CPU
	C2	LUT method by using “Scatter” with a CPU
	C3	Direct calculation method of Eq. (1) by using “Gather” with a CPU
GPU	G1	Direct calculation method of Eq. (1) by using “Scatter” with a GPU
	G2	LUT method by using “Scatter” with a GPU
	G3	Direct calculation method of Eq. (1) by using “Gather” with a GPU



(a) Passing region of light. (b) Operation block.

Fig. 4: Accumulating calculation in GPU implementation.

4 RESULT

We used a 3D object consisting of 978,416 points. We changed the number of object points N by randomly sampling the 3D object [11], and we evaluated the calculation times between the 3D object and WRP. Prior to the calculation, we sorted coordinate information of the 3D object to increase the locality of the object points, leading to an improvement of the cache memory efficiency.

First, we consider the results of each CPU implementation as shown in Table 1. Figure 5 shows the relationship between the number of object points and the calculation time. We confirmed that the implementation of C2 (the combination of the “Scatter” and the LUT method) achieved the shortest calculation time among the three implementations. Because the cache memory is well work for this small amount of the LUT, the implementation of C2, which has a small number of arithmetic operations compared to the implementations of C1 and C3, had the shortest calculation time among the three implementations.

Next, we consider the results of the GPU implementation as shown in Table 1. Figure 6 shows the relationship between the number of object points and the calculation time. We confirmed that the implementations of G1 and G3 by the direct calculation method have shorter calculation times than that of G2 by the LUT method. This is a different result in comparison with the CPU implementation. The calculation speed of arithmetic operations in a GPU is generally faster than the access speed to the global memory on a GPU board. In the implementation of G2, the complex amplitude is read from the LUT memory stored in the global memory. Therefore, the implementation of G2 by the LUT method took longer calculation time than those of G1 and G3 by the direct calculation method of Eq. (1).

Table2 shows the calculation times between the 3D objects and WRP in the number of object points $N = 163{,}840$. We confirmed that the LUT method has a lower speed-up ratio than the direct calculation method because most of the calculation times occupied by memory access.

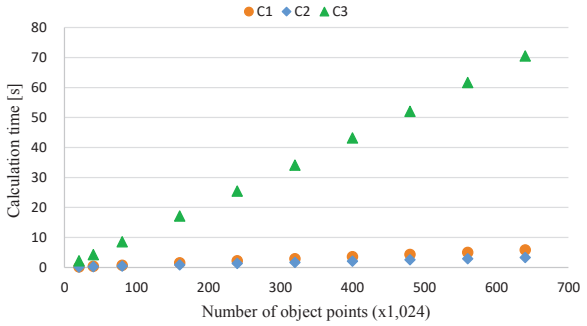


Fig. 5: Relationship between the number of object points and the calculation time with a CPU.

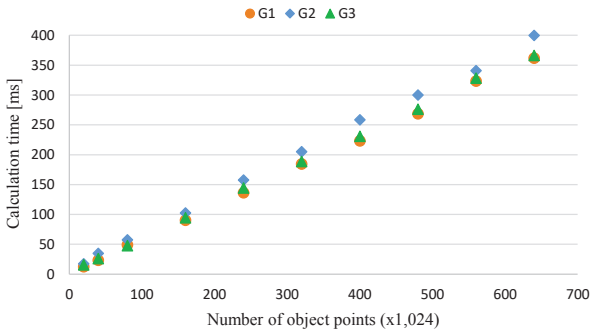


Fig. 6: Relationship between the number of object points and the calculation time with a GPU.

Table 2: Calculation time between the 3D objects and WRP ($N = 163,840$).

Symbol	Calculation time [ms]
C1	1,561
C2	866
C3	17,123
G1	90
G2	102
G3	95

5 CONCLUSIONS

In this study, we confirmed that the CPU implementation by using the combination of the "Scatter" and the LUT method is better than other implementations, and the GPU implementation by using the direct calculation method is better than those by using the LUT method. In future works, by changing the distance from the 3D objects to WRP, we will evaluate calculation times changing the distance from the 3D objects to WRP.

ACKNOWLEDGMENT

This work was partially supported by JSPS KAKENHI Grant Numbers 19H04132 and 19H01097.

REFERENCES

[1] G. Lippmann "Epreuves reversible. Photographies integrals," C. R. Acad. Sci. **146**, 446-451 (1908).

[2] S. Lee, J. Park, J. Heo, B. Kang, D. Kang, H. Hwang, J. Lee, Y. Choi, K. Choi, and D. Nam, "Autostereoscopic 3D display using directional subpixel rendering," Opt. Express **26**, 20233-20247 (2018).

[3] P. S. Hilaire, S. A. Benton, M. Lucente, M. L. Jepsen, J. Kollin, H. Yoshikawa, and J. Underkoffle, "Electronic display system for computational holography," Proc. SPIE **1212**, 174-182 (1990).

[4] Z. Zhang, C. P. Chen, Y. Li, B. Yu, L. Zhou, and Y. Wu, "Angular multiplexing of holographic display using tunable multi-stage gratings," Mol. Cryst. Liq. Cryst. 657, 102-106 (2017).

[5] H. Yanagihara, T. Kakue, T. Shimobaba, and T. Ito, "Real-time three-dimensional video reconstruction of real scenes with deep depth using electro-holographic display system," Opt. Express **27**, 15662-15678 (2019).

[6] T. Sugie, T. Akamatsu, T. Nishitsuji, R. Hirayama, N. Masuda, H. Nakayama, Y. Ichihashi, A. Shiraki, M. Oikawa, N. Takada, Y. Endo, T. Kakue, T. Shimobaba, and T. Ito, "High-performance parallel computing for next-generation holographic imaging," Nature Electron. **1**, 254-259 (2018).

[7] Y. Pan, X. Xu, S. Solanki, X. Liang, R. B. A. Tanjung, C. Tan, and T. C. Chong, "Fast CGH computation using S-LUT on GPU," Opt. Express **17**, 18543-18555 (2009).

[8] T. Shimobaba, N. Masuda, and T. Ito, "Simple and fast calculation algorithm for computer-generated hologram with wavefront recording plane," Opt. Lett. **34**, 3133-3135 (2009).

[9] Visual Studio, https://visualstudio.microsoft.com.

[10] CUDA, https://developer.nvidia.com/cuda-zone.

[11] H. Yanagihara, T. Kakue, T. Shimobaba, and T. Ito, "Real-Time 3D Image Reconstruction System of Real-Scenes by Electro-Holography using 3D Objects Acquired with Kinect," Proceedings of The 10th International Conference on 3D Systems and Applications, HT-4 (2018).