Real-time spatiotemporal division multiplexing electroholography for 1,200,000 object points using multiple-graphics processing unit cluster

Hiromi Sannomiya[1], Naoki Takada[2,*], Kohei Suzuki[1], Tomoya Sakaguchi[1], Hirotaka Nakayama[3], Minoru Oikawa[2], Yuichiro Mori[2], Takashi Kakue[4], Tomoyoshi Shimobaba[4], and Tomoyoshi Ito[4]

[1]*Graduate School of Integrated Arts and Sciences, Kochi University, Kochi 780-8520, Japan*
[2]*Research and Education Faculty, Kochi University, Kochi 780-8520, Japan*
[3]*National Astronomical Observatory of Japan, Mitaka 181-8588, Japan*
[4]*Graduate School of Engineering, Chiba University, Inage-ku 263-8522, Japan*
**Corresponding author: ntakada@is.kochi-u.ac.jp*

Received December 30, 2019; accepted April 30, 2020; posted online June 12, 2020

Computationally, the calculation of computer-generated holograms is extremely expensive, and the image quality deteriorates when reconstructing three-dimensional (3D) holographic video from a point-cloud model comprising a huge number of object points. To solve these problems, we implement herein a spatiotemporal division multiplexing method on a cluster system with 13 GPUs connected by a gigabit Ethernet network. A performance evaluation indicates that the proposed method can realize a real-time holographic video of a 3D object comprising ∼1,200,000 object points. These results demonstrate a clear 3D holographic video at 32.7 frames per second reconstructed from a 3D object comprising 1,064,462 object points.

Keywords: real-time electroholography; multiple-graphics processing unit cluster; graphics processing unit; spatiotemporal division multiplexing electroholography.
doi: 10.3788/COL202018.070901.

Real-time electroholography based on computer-generated holograms (CGHs) is expected to become the ultimate three-dimensional (3D) television[1,2]. However, computationally, the CGH calculation rapidly becomes prohibitively expensive because real-time electroholography requires processing extremely high floating-point arithmetic. The image quality of holographic video deteriorates when reconstructed from a point-cloud model comprising a huge number of object points. Two proposals to suppress this deterioration are the time multiplexing for two-dimensional reconstruction[3] and the spatiotemporal division multiplexing for clear 3D holographic video playback[4]. Large-scale electroholography using the spatiotemporal division multiplexing approach[4] implemented on the Horn-8 system has been reported[5].

A modern graphics processing unit (GPU) is a cost-effective processor capable of high-floating-point arithmetic processing and fast computer-graphics processing. Thus, GPUs accelerate CGH calculations and directly display the calculated CGH on a spatial light modulator (SLM)[6–14]. Conversely, spatiotemporal division multiplexing uses moving image features[15]. This approach accelerates CGH calculations several-fold.

A PC cluster consisting of multiple PCs with multiple GPUs is called a multi-GPU cluster and can be used to significantly accelerate large-pixel-count CGH calculations[16–20]. Reference [16] directly connected the GPUs of a multi-GPU cluster to multiple SLMs to show that a multi-GPU cluster is suitable for real-time electroholography involving a large-pixel-count CGH. However, such a multi-GPU cluster system with multiple SLMs is very expensive. Real-time electro-holography using a multi-GPU cluster with a single SLM is low cost, but requires the CGH data transfer between the nodes, which prevents real-time electroholography. To address this problem, we used a high-speed InfiniBand network in a multi-GPU cluster system and applied this system to real-time electroholography[21] and fast time-division color electroholography[22]. We also realized real-time color electroholography by using a multi-GPU cluster system with three SLMs combined with an InfiniBand network[23]. Furthermore, we proposed a packing and unpacking method to reduce CGH data transfer between the nodes of the multi-GPU cluster[24]. We demonstrated real-time electroholography by using a multi-GPU cluster with 13 GPUs (NVIDIA GeForce 1080 Ti) connected by a gigabit Ethernet and a single SLM.

In this Letter, we propose clear real-time electroholography based on spatiotemporal division multiplexing using moving image features and a multi-GPU cluster system connected by a gigabit Ethernet network. The proposed method does not use cache memory.

In previous work[4,15], we proposed two types of spatiotemporal division multiplexing. The first method suppresses the deterioration of 3D holographic video reconstructed from a point-cloud model comprising a huge number of object points (Fig. 1)[4]. The second method uses moving image features to accelerate the CGH calculation (Fig. 2)[15]. In both methods, a 3D object is divided into several objects

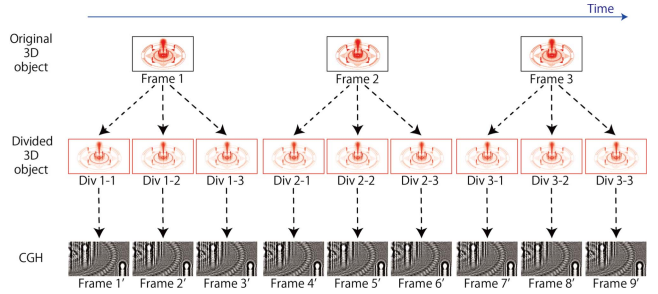

Fig. 1. Spatiotemporal division multiplexing approach for suppressing the deterioration of a 3D holographic video reconstructed from a point-cloud model comprising a huge number of object points.

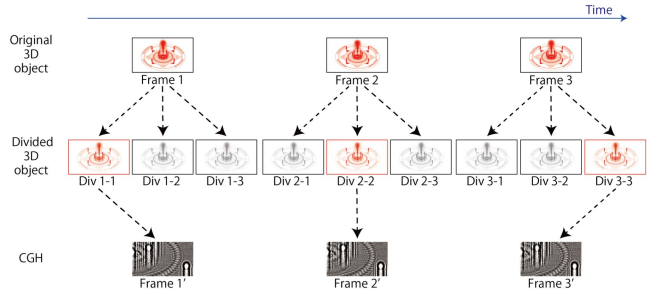

Fig. 2. Spatiotemporal division multiplexing approach using moving image features.

at each frame of the original 3D video. Figures 1 and 2 show examples in which the original 3D object is divided into three objects in each frame, with the divided objects labeled Div $i-1$, Div $i-2$, and Div $i-3$ for frame i.

In the spatiotemporal division multiplexing method for suppressing the deterioration of 3D holographic video, all the divided objects are used in each frame. At frame i shown in Fig. 1, the CGHs are generated from the divided objects Div $i-1$, Div $i-2$, and Div $i-3$, and the CGHs display sequentially on an SLM. Here, the reconstructed 3D holographic video has three times as many frames as the original 3D video. This approach requires three times more time than the original 3D video reconstruction.

As shown in Fig. 2, the spatiotemporal division multiplexing approach using the moving image features uses only one of the divided objects. Here, a different divided object is selected for every three frames. In each frame, the number of object points contributing to one divided object is one-third of that contributing to the original 3D object. Thus, the subsequent CGH calculation is three times faster than that using the original 3D video. However, long CGH calculations for each frame prevent smooth real-time reconstruction of moving 3D images. As a result, we have never applied the spatiotemporal multiplexing approach using moving image features to a point-cloud model comprising a huge number of object points.

In the spatiotemporal multiplexing approach using moving image features, the equation used for the CGH calculations[6] is

$$I(x_h, y_h, 0) = \sum_{j=1}^{N_p} A_j \cos\left\{\frac{\pi}{\lambda z_j}[(x_h - x_j)^2 + (y_h - y_j)^2]\right\}, \tag{1}$$

where $(x_h, y_h, 0)$ are the coordinates of a point on the CGH, (x_j, y_j, z_j) and A_j are the coordinates of the jth object point in the 3D-object-based point-cloud model, N_p is the total number of object points in the 3D object, and λ is the wavelength of the reconstructing light. Note that the Fresnel approximation is used in Eq. (1).

The value calculated from Eq. (1) for each point in the CGH is binarized by using a threshold value of zero[25]. The binary CGH is generated from the binarized value for each point in the original CGH. The CGH calculation time increases proportionally to the number of the object points. Because it is difficult to realize real-time electroholography for a point-cloud model comprising a huge number of object points, we have restricted ourselves to a 1920 × 1024 binary CGH.

In the 3D model "fountain" comprising 1,064,462 object points, we investigated the number of space divisions for spatiotemporal division multiplexing using moving image features. The 3D model was located 1.5 m away from the CGH. The size of the 3D model is approximately 70 mm × 50 mm × 50 mm. The 3D model is divided into several objects, and the CGHs are generated from the divided objects. All CGHs are repeatedly displayed on an SLM. For the SLM, we used a liquid-crystal display panel extracted from a projector (EMP-TW1000, Epson, Inc.). A green (532 nm) semiconductor laser was used for reconstruction. Figure 3 shows the reconstructed 3D images; the clearest images were obtained with six space divisions.

We used the multi-GPU cluster system shown in Fig. 4. A gigabit Ethernet network connected the multi-GPU cluster system, which consisted of a CGH display node and four CGH calculation nodes. The CGH display node had a GPU, and each of the CGH calculation nodes had three GPUs, for a total of 16 GPUs. Each GPU was a NVIDIA GeForce GTX 1080 Ti (see Table 1 for specifications of each node in the multi-GPU cluster system). The CGH display node also plays the role of server for the network file system (NFS). Figure 5 shows the pipeline processing executed on the multi-GPU cluster system. The frames from Frame 1′ to Frame 12′ shown in Fig. 2 are assigned to GPUs 1 to 12, respectively, on the four CGH calculation nodes. In Fig. 5, the actual CGH calculation time for each single frame is equal to the twelve-fold value of the display-time interval T because the total number of GPUs in the CGH calculation nodes is twelve. The CGH calculation time is proportional to the number of 3D-object points. The GPUs use Eq. (1) to generate the CGH data from the divided 3D objects of the assigned frames. The computational complexity of Eq. (1) becomes enormous. However, in the CGH calculation using Eq. (1), the actual computational performance of the GPU is related to not only the computational complexity but also

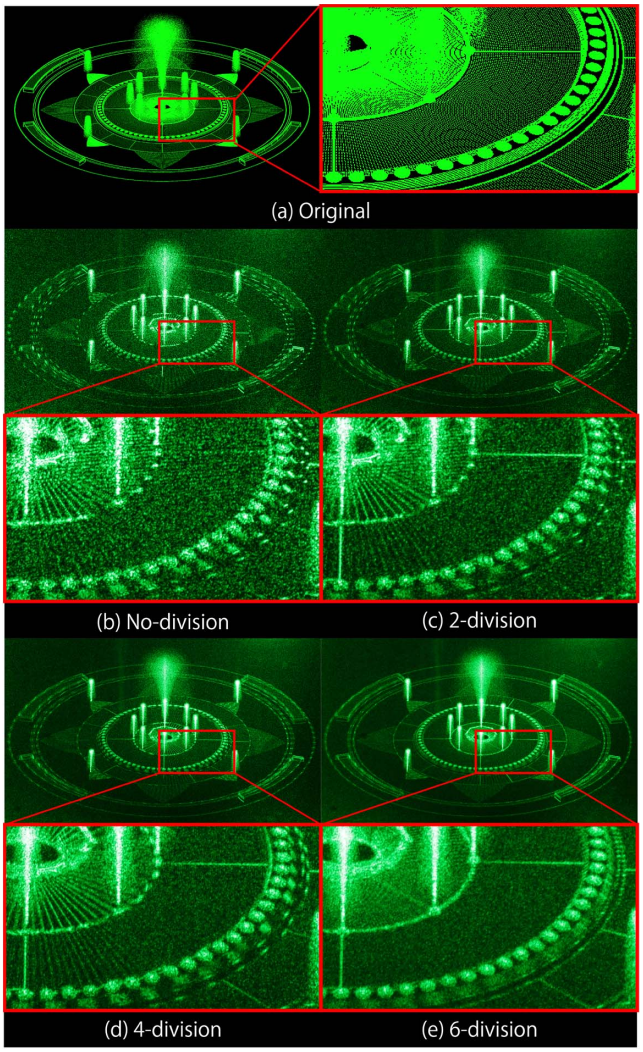

Fig. 3. Reconstructed 3D image from a 3D object "fountain" comprising 1,064,462 object points.

the number of the data accesses to the off-chip memory on the GPU[26]. Furthermore, the performance of the GPU computation is remarkably reduced when the number of data accesses is very large compared with the amount of CGH calculations. The optimized method[16] can reduce the number of the data accesses to the off-chip memory and provide the high-speed CGH computation. Therefore, we used the optimized method[16] in the CGH calculation.

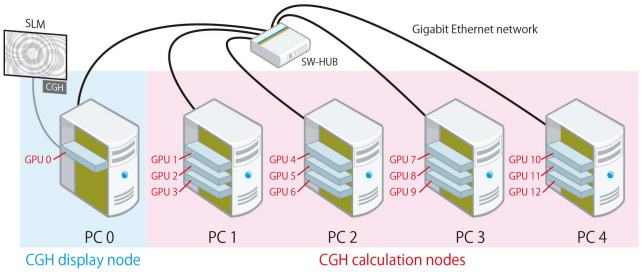

Fig. 4. Multi-GPU cluster system with multiple GPUs connected by a gigabit Ethernet network and a single SLM.

Table 1. Specifications of Each Node in the Multi-GPU Cluster System

CPU	Intel Core i7 7800X (clock speed: 3.5 GHz)
Main memory	DDR4-2666 16 GB
OS	Linux (CentOS 7.6 x86_64)
Software	NVIDIA CUDA 10.1 SDK, OpenGL, MPICH 3.2
GPU	NVIDIA GeForce GTX 1080 Ti

The packed CGH data are generated by the packing processing and sent to the CGH display node. In the CGH display node, a GPU unpacks the packed CGH data and generates the binary CGHs. The binary CGHs are displayed sequentially on the liquid-crystal display panel connected to the CGH display node. Here, the packing and unpacking serve to reduce the CGH transfer data[24]. These processes are then repeated until reaching the last frame of the 3D video.

The time required to read the coordinate data of the object points from auxiliary storage becomes non-negligible when the number of the 3D-object points is huge. We investigated the total time required to display twelve-frame sequences because, by using pipeline processing, all GPUs of the CGH calculation nodes generated twelve CGH data in each cycle. In each of the CGH calculation nodes, we used two codes for serial computing [see Fig. 6(a)] and for parallel computing [see Fig. 6(b)]. The object data in the process "read object data," which means to read object data from the NFS server, are the coordinates of the object points expressed as binary data. Fig. 7 shows the total display time for sets of twelve frames when using the serial computing scheme shown in Fig. 6(a) and when using the parallel computing scheme shown in Fig. 6(b) for 1,200,000 object points. Here, no cache memory was used when reading the coordinate data. Twelve CGHs for twelve frames were calculated by using twelve GPUs on the CGH calculation nodes. In Fig. 7, "SSD" and "HDD" refer to a solid-state drive and a hard disk drive, respectively, on the

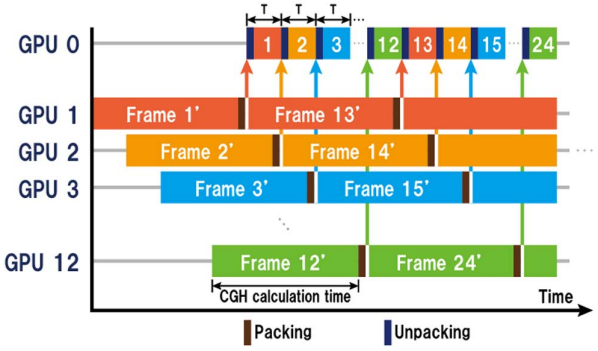

Fig. 5. Pipeline processing for the spatiotemporal electroholography system shown in Fig. 2.

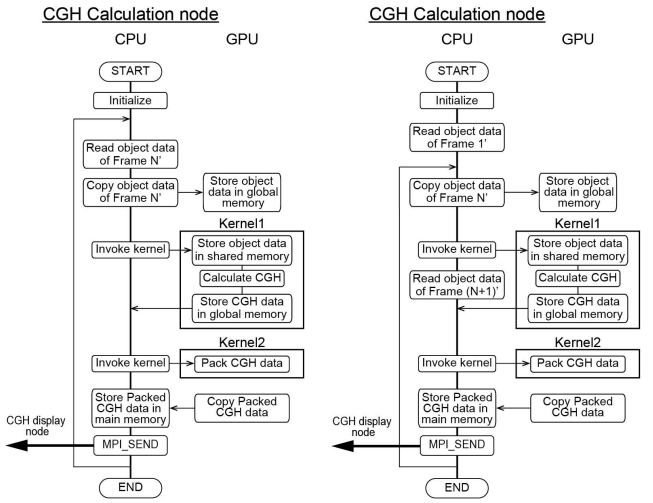

Fig. 6. Read data processing and CGH calculation on each CGH calculation node in the multi-GPU cluster system shown in Fig. 4. (a) Serial computing. (b) Parallel computing.

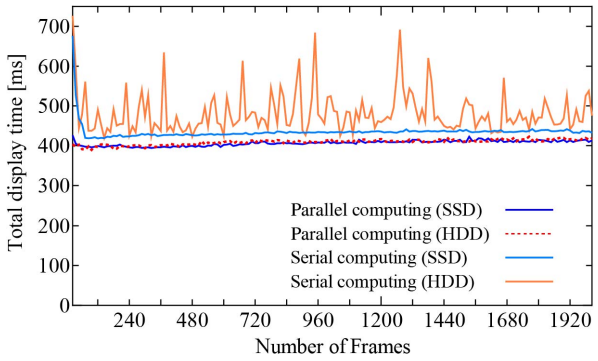

Fig. 7. Comparison of the total display time for every 12 frames using serial computing shown in Fig. 6(a) with that using parallel computing shown in Fig. 6(b) when the number of object points is 1,200,000.

NFS server to store the coordinates of the object points. We used a Western Digital WD20EZAZ-RT (2 TB) HDD and an Intel Optane 900P (280 GB) SSD. The result shown in Fig. 7 indicates that the serial computing outlined in Fig. 6(a) is substantially affected by HDD access time when the HDD serves as the storage for the NFS server. When using parallel computing [Fig. 6(b)], the time required to read the object-point coordinates is completely hidden within the time required to do each CGH calculation using a GPU from the CGH calculation nodes, regardless of whether the HDD or SSD is used.

Figure 8 plots the display-time interval T shown in Fig. 5 versus the number of object points when implementing spatiotemporal division multiplexing using moving image features on the multi-GPU cluster system shown in Fig. 4. Here, we use six space divisions. The display-time interval T increases in proportion with the number of object points. The display-time interval T is 34.6 ms for 1,200,000 object points, and the frame rate is 28.9 frames

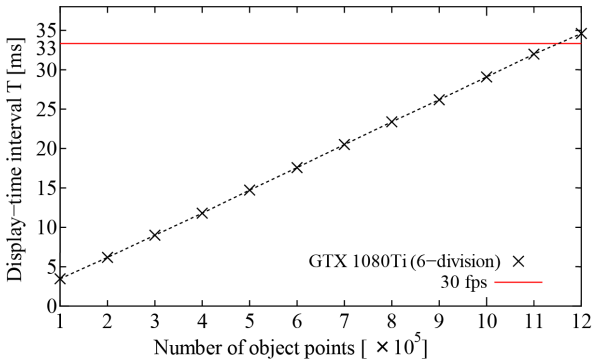

Fig. 8. Display-time interval T shown in Fig. 5 plotted versus the number of object points when using the spatiotemporal division multiplexing approach using moving image features implemented on the multi-GPU cluster system shown in Fig. 4.

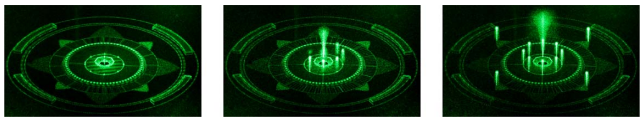

Fig. 9. Snapshot of a reconstructed 3D video (Video 1).

per second (fps). Figure 8 shows the performance of the proposed method. The proposed method provides clear real-time 3D holographic video for the 3D model comprising a huge number of object points. Therefore, Fig. 8 does not always show that the proposed method requires a high refresh rate of SLM.

Figure 9 shows snapshots of the reconstructed 3D video (Video 1) from the original 3D video "fountain" comprising 1,064,462 object points and with six space divisions. Table 2 lists the frame rate of the reconstructed 3D video from the original 3D video "fountain" comprising 1,064,462 object points and for the number of space divisions. We obtained a clear holographic 3D video reconstructed from a 3D object comprising 1,064,462 object points at 32.7 fps with six space divisions.

In conclusion, we implemented the spatiotemporal multiplexing approach using moving image features on a multi-GPU cluster system with 13 GPUs. A performance evaluation indicates that the proposed method can realize

Table 2. Frame Rate of the Reconstructed 3D Video from the Original 3D Video "Fountain" Comprising 1,064,462 Object Points Against the Number of Space Divisions

Number of Space Divisions	Object Points	Frame Rate (fps)
No division	1,064,462	5.43
Two divisions	532,231	10.86
Four divisions	266,116	21.70
Six divisions	177,411	32.70

a real-time holographic video of a 3D object comprising approximately 1,200,000 object points. We obtained a clear real-time spatiotemporal holographic 3D video of a 3D object comprising 1,064,462 object points. The proposed method facilitates the handling of the clear real-time 3D holographic video, is applicable to various algorithms for the CGH calculation, and thereby significantly contributes to the development of the ultimate holographic 3D television.

This work was partially supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI (Nos. 18K11399 and 19H01097) and the Telecommunications Advancement Foundation.

References

1. S. A. Benton and J. V. M. Bove, *Holographic Imaging* (Wiley, 2008).
2. 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, Nat. Electron. **1**, 254 (2018).
3. Y. Mori, T. Fukuoka, and T. Nomura, Appl. Opt. **53**, 8182 (2014).
4. N. Takada, M. Fujiwara, C. W. Ooi, Y. Maeda, H. Nakayama, T. Kakue, T. Shimobaba, and T. Ito, IEICE Trans. Electron. **E100.C**, 978 (2017).
5. Y. Yamamoto, H. Nakayama, N. Takada, T. Nishitsuji, T. Sugie, T. Kakue, T. Shimobaba, and T. Ito, Opt. Express **26**, 34259 (2018).
6. N. Masuda, T. Ito, T. Tanaka, A. Shiraki, and T. Sugie, Opt. Express **14**, 603 (2006).
7. A. Shiraki, N. Takada, M. Niwa, Y. Ichihashi, T. Shimobaba, N. Masuda, and T. Ito, Opt. Express **17**, 16038 (2009).
8. Y. Pan, X. Xu, S. Solanki, X. Liang, R. B. A. Tanjung, C. Tan, and T.-C. Chong, Opt. Express **17**, 18543 (2009).
9. P. Tsang, W. K. Cheung, T.-C. Poon, and C. Zhou, Opt. Express **19**, 15205 (2011).
10. J. Weng, T. Shimobaba, N. Okada, H. Nakayama, M Oikawa, N. Masuda, and T. Ito, Opt. Express **20**, 4018 (2012).
11. G. Li, K. Hong, J. Yeom, N. Chen, J.-H. Park, N. Kim, and B. Lee, Chin. Opt. Lett. **12**, 060016 (2014).
12. Z. Chen, X. Sang, Q. Lin, J. Li, X. Yu, X. Gao, B. Yan, C. Yu, W. Dou, and L. Xiao, Chin. Opt. Lett. **14**, 080901 (2016).
13. Y. Zhang, J. Liu, X. Li, and Y. Wang, Chin. Opt. Lett. **14**, 030901 (2016).
14. D.-W. Kim, Y.-H. Lee, and Y.-H. Seo, Appl. Opt. **57**, 3511 (2018).
15. H. Niwase, N. Takada, H. Araki, H. Nakayama, A. Sugiyama, T. Kakue, T. Shimobaba, and T. Ito, Opt. Express **22**, 28052 (2014).
16. N. Takada, T. Shimobaba, H. Nakayama, A. Shiraki, N. Okada, M. Oikawa, N. Masuda, and T. Ito, Appl. Opt. **51**, 7303 (2012).
17. Y. Pan, X. Xu, and X. Liang, Appl. Opt. **52**, 6562 (2013).
18. B. J. Jackin, H. Miyata, T. Ohkawa, K. Ootsu, T. Yokota, Y. Hayasaki, T. Yatagai, and T. Baba, Opt. Lett. **39**, 6867 (2014).
19. B. J. Jackin, S. Watanabe, K. Ootsu, T. Ohkawa, T. Yokota, Y. Hayasaki, T. Yatagai, and T. Baba, Appl. Opt. **57**, 3134 (2018).
20. T. Baba, S. Watanabe, B. J. Jackin, K. Ootsu, T. Ohkawa, T. Yokota, Y. Hayasaki, and T. Yatagai, IEICE Trans. Inf. Sys. **E102.D**, 1310 (2019).
21. H. Niwase, N. Takada, H. Araki, Y. Maeda, M. Fujiwara, H. Nakayama, T. Kakue, T. Shimobaba, and T. Ito, Opt. Eng. **55**, 093108 (2016).
22. H. Araki, N. Takada, S. Ikawa, H. Niwase, Y. Maeda, M. Fujiwara, H. Nakayama, M. Oikawa, T. Kakue, T. Shimobaba, and T. Ito, Chin. Opt. Lett. **15**, 120902 (2017).
23. S. Ikawa, N. Takada, H. Araki, H. Niwase, H. Sannomiya, H. Nakayama, M. Oikawa, Y. Mori, T. Kakue, T. Shimobaba, and T. Ito, Chin. Opt. Lett. **18**, 010901 (2020).
24. H. Sannomiya, N. Takada, T. Sakaguchi, H. Nakayama, M. Oikawa, Y. Mori, T. Kakue, T. Shimobaba, and T. Ito, Chin. Opt. Lett. **18**, 020902 (2020).
25. W.-H. Lee, Appl. Opt. **18**, 3661 (1979).
26. A. Waterman and D. Patterson, Commun. ACM **52**, 65 (2009).