Optical Engineering

OpticalEngineering.SPIEDigitalLibrary.org

Two schemes for rapid generation of digital video holograms using PC cluster

Hanhoon Park
Joongseok Song
Changseob Kim
Jong-Il Park

SPIE.

Hanhoon Park, Joongseok Song, Changseob Kim, Jong-Il Park, "Two schemes for rapid generation of digital video holograms using PC cluster," *Opt. Eng.* **56**(12), 123104 (2017), doi: 10.1117/1.OE.56.12.123104.

Two schemes for rapid generation of digital video holograms using PC cluster

Hanhoon Park,[a] Joongseok Song,[b] Changseob Kim,[c] and Jong-Il Park[c,*]
[a]Pukyong National University, Department of Electronic Engineering, Busan, Republic of Korea
[b]Kohyoung Technology, Vision Team, Software Laboratory, Seoul, Republic of Korea
[c]Hanyang University, Department of Computer Software, Seoul, Republic of Korea

Abstract. Computer-generated holography (CGH), which is a process of generating digital holograms, is computationally expensive. Recently, several methods/systems of parallelizing the process using graphic processing units (GPUs) have been proposed. Indeed, use of multiple GPUs or a personal computer (PC) cluster (each PC with GPUs) enabled great improvements in the process speed. However, extant literature has less often explored systems involving rapid generation of multiple digital holograms and specialized systems for rapid generation of a digital video hologram. This study proposes a system that uses a PC cluster and is able to more efficiently generate a video hologram. The proposed system is designed to simultaneously generate multiple frames and accelerate the generation by parallelizing the CGH computations across a number of frames, as opposed to separately generating each individual frame while parallelizing the CGH computations within each frame. The proposed system also enables the subprocesses for generating each frame to execute in parallel through multithreading. With these two schemes, the proposed system significantly reduced the data communication time for generating a digital hologram when compared with that of the state-of-the-art system.
 [DOI: 10.1117/1.OE.56.12.123104]

Keywords: computer-generated holography; graphic processing unit-equipped PC cluster; CUDA; digital video hologram; data communication time; multithreading.

Paper 171300 received Aug. 18, 2017; accepted for publication Nov. 9, 2017; published online Dec. 11, 2017.

1 Introduction

Holography is a technology that enables people to view three-dimensional (3-D) images (called holographic images or simply holograms) displayed in real space with the naked eye. Although a hologram was originally generated using optical apparatuses,[1] it can be digitally implemented on computers with many advantages.[2,3] Computer-generated holography (CGH) is a method that computes digital holographic interference patterns required for generating holograms in a holographic 3-D display. There are mainly two types of CGHs, point-based and Fourier-transform-based (also called polygon-based).[4–6] Both generally involve a huge amount of computations; thus, computational reduction has been a main research topic in this field. However, the point-based method further suffers from the high computational complexity as shown in Eq. (1) wherein the computational complexity rapidly increases in proportion to the hologram resolution and the number of light sources (referring to pixels with a nonzero intensity value in a depth image) of a 3-D object.

$$I(x_h, y_h) = \sum_{l}^{L} A_l(x_l, y_l, z_l) \cos\left(\frac{2\pi z_l}{\lambda} + \frac{\pi p^2 D}{\lambda z_l}\right), \tag{1}$$

where $D = (x_h - x_l)^2 + (y_h - y_l)^2$, $0 \le x_h \le W_h - 1$, and $0 \le y_h \le H_h - 1$. Here, I and A denote the light intensities of a hologram and a 3-D object (or a set of 3-D light sources), respectively. x_l, y_l, and z_l are the 3-D coordinates of the light sources. In addition, λ and p denote the wavelength of the reference wave and pixel pitch, respectively, and L denotes the number of 3-D object light sources. W_h and H_h are the width and height of the hologram.

Several software-based[5–12] and hardware-based[13–19] methods were proposed to reduce the computational complexity. Software-based methods have tried to store the CGH computation results in a look-up table in advance,[7,8] recursively generate the intensities of the rest using the precalculated values of neighbor or particular CGH pixels,[9,10] and reduce the CGH computation using a cosine approximation algorithm,[11] an effective diffraction area recording method,[12] a layered model,[5] or a patch model.[6] However, those could not speed up enough to generate high-resolution holograms in real time, and some of them have degraded the quality of holograms. Conversely, hardware-based methods have generated high-resolution holograms in near real-time without any quality change by parallelizing the CGH computation using field-programmable gate array,[13] a single unit or multiple graphic processing units (GPUs),[14–18] and even a personal computer (PC) cluster system[19,20] composed of multiple PCs in which each PC has multiple GPUs. As a state-of-the-art method, a scalable and flexible PC cluster system was proposed[21] to generate higher resolution holograms [called high-quality (HQ) holograms] with a considerably larger number of object light sources. The system was a server-client system and could be flexibly composed of different numbers or performance of PCs and GPUs. A PC acted as a server and periodically investigated the computing power of each client PC and optimally distributed the amount of computations. Consequently, the system generated an HQ hologram (1536 × 1536 resolution and more than 2.1 million

*Address all correspondence to: Jong-Il Park, E-mail: jipark@hanyang.ac.kr

light sources) in 10 s. This was highly efficient when compared with the previous systems. However, the method still involved a significantly long period of time to generate an HQ hologram even if the cluster system was used. Hence, it is important to further improve the performance of the cluster system. In particular, the server-client system spent a considerable amount of time communicating data between the server and the clients.[21] Therefore, a method for reducing the communication time is necessary, and this is the main focus of this study.

Digital video holograms are composed of a number of frames, and each frame can be generated separately and quickly using the aforementioned existing methods or systems as in Refs. 21 and 20. Indeed, this has been a common way to generate digital video holograms. Strictly speaking, there has been no specialized approach for generating video holograms in the literature. However, it is possible to further reduce the hologram generation time (exactly, the data communication time between the server and the clients) by considering and generating the frames together. In this context, instead of distributing/parallelizing the CGH computations for generating each individual frame, this study proposes assigning all the computations of a single frame to a single PC and determining the number of frames assigned to each PC on the basis of the performance of each PC. This implies that the parallelization is achieved on a frame-to-frame basis (Scheme 1). In addition, the previous studies that focused on fast generation of a single hologram paid no attention to the parallelization of subprocesses (i.e., distribution, CGH computation, and collection, which will be specified later) for hologram generation because the subprocesses should be executed sequentially for the generation of a single hologram. However, they can be parallelized in video hologram generation, and this parallelization can reduce the data communication time. Therefore, this study proposes parallelizing the subprocesses and provides a practical solution based on multithreading (Scheme 2). With these two schemes, the data communication time between the server and the clients can be minimized.

The first scheme is similar to that in a previous study[20] in that a client PC is fully in charge of the CGH computations of a frame. However, in the study,[20] the framewise generation was not newly designed for quick generation of video holograms and the system required all the clients to have the same performance (i.e., identical GPUs), which is usually not the case in real computing environments. In addition, the data transmission time between the server and the client PCs was ignored using an extremely-high-speed network. Our second scheme provides a practical solution to reduce the data transmission time in real network environments.

2 Proposed System: A Digital Video Hologram Generation System with Two Speedup Schemes

The proposed system is very similar to that used in a previous study.[21] Both systems are based on the server-client architecture, where the client PCs have different performance; thus, the server PC periodically investigates the time varying computing power (s) of each client PC by sending a small and identical amount of CGH computations to each client and receiving the computation time (T_{ct}) measured by each client. The computing power is computed as follows:

$$s = \kappa / T_{ct}. \tag{2}$$

Here, κ is a predefined constant. For the true generation of digital holograms, the server PC assigns a certain amount of CGH computations in proportion to the computing power of each client PC (called distribution subprocess hereafter). Each client PC performs the assigned CGH computations (called CGH computation subprocess hereafter). Then, the server collects the results from each client and generates the final holograms by accumulating/arranging them (called collection subprocess hereafter). However, in the previous study,[21] generation of each frame ($W_h \times H_h$) of a video

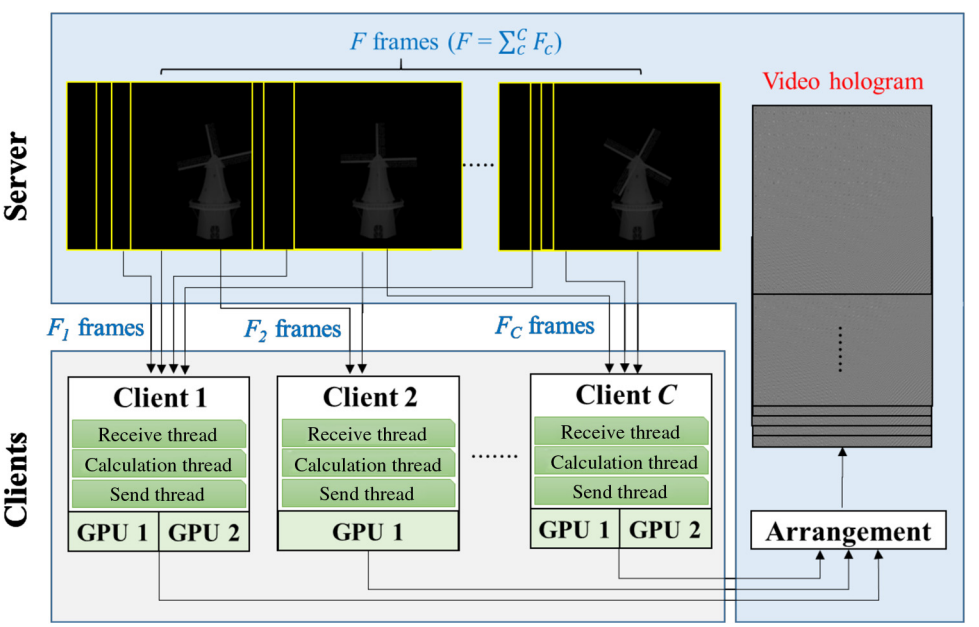

Fig. 1 Overview of the proposed system.

hologram was parallelized separately. That is, the light sources for generating a single frame were distributed to C clients and the partial CGH computations with the distributed light sources were performed for each client. Then, the intermediate interference patterns (with the same resolution as the final hologram, i.e., $W_h \times H_h$) computed for each client were sent back to the server and were accumulated. Therefore, given that the mean data communication time between the server and each client was T_t, the total communication time for collecting the results from the clients was CT_t (the distribution time could be ignored when compared with the collection time). With the high hologram resolution and the large number of PCs, the communication time was too long, and this presented a significant challenge for the rapid generation of each frame. In the generation of a video hologram, the same process was repeated for each frame. The total generation time linearly increased in proportion to the number of frames F; hence, the total communication time was CT_tF. The proposed system tries to reduce the data communication time in two ways. The overview of the proposed system is shown in Fig. 1.

2.1 *Distribution of Computations on a Frame-to-Frame Basis*

The proposed system distributes a certain number of frames to each PC on the basis of its performance of each PC as described in Eq. (3) (also see Fig. 2). It assigns all of the CGH computations to generate each frame for a client; thus, the data communication time to generate each frame is T_t and not CT_t (once for each frame, the fully generated hologram is sent to the server). In other words, the proposed system can reduce the data communication time by a factor of C. The total communication time is T_tF during the generation of a video hologram.

$$\Psi_c = \left\lceil \Psi\left(s_c / \sum_{c}^{C} s_c \right) \right\rceil. \tag{3}$$

Here, Ψ and Ψ_c denote the number of frames that is a controllable throughput ($\leq F$) and is allocated to each client, respectively. s_c denotes the computing power of the c'th client.

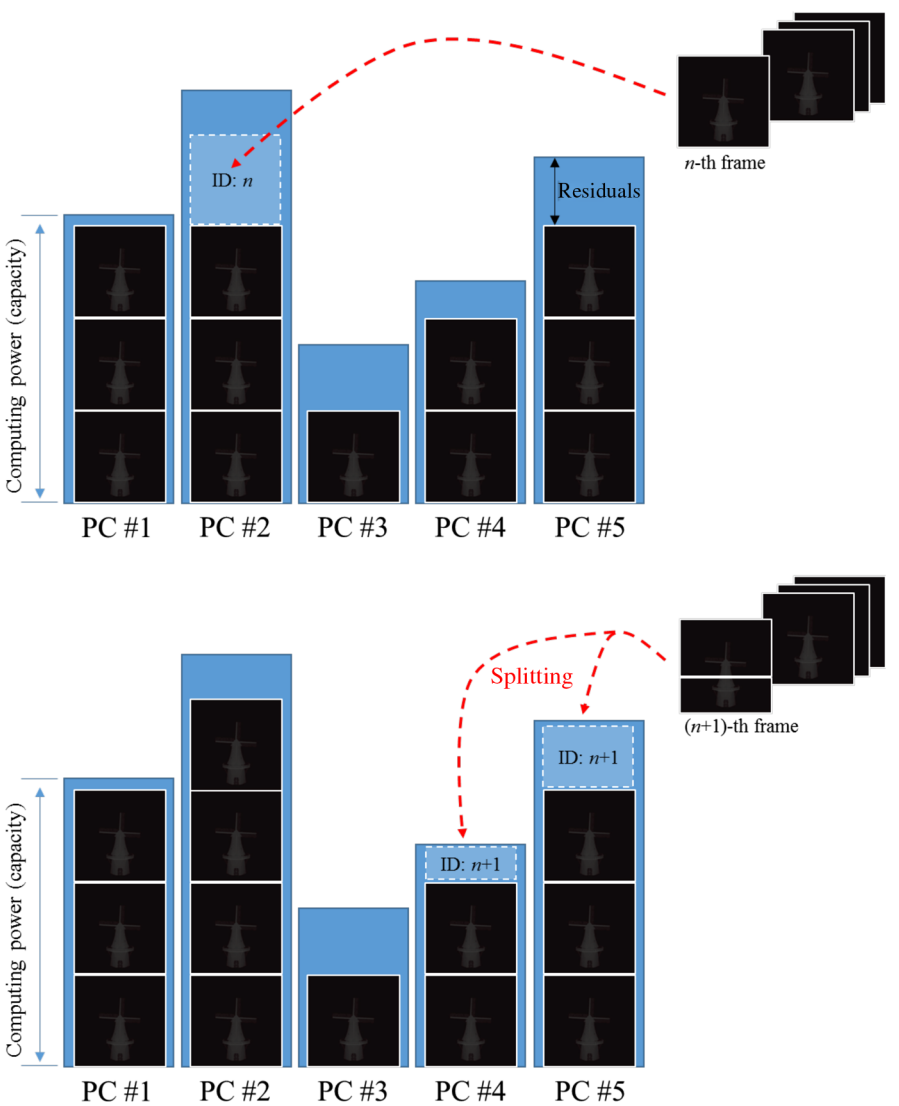

Fig. 2 Parallelizing the CGH computation on a frame-to-frame basis (upper one) and on a light source basis (lower one). The height of the blue boxes indicates the amount of computations that may be processed at a time on each PC.

Conversely, the CGH computation time (denoted by T_{cp}) of each frame in the proposed system can be high since all the computations are performed on a single PC. This is in contrast with the method used in the previous study in which the computations were distributed to multiple PCs.[21] However, with respect to a video hologram with a large number of frames, multiple frames can be simultaneously generated via parallel processing with multiple PCs. Additionally, the computation time is optimally minimized by determining the number of frames generated in each PC based on the computing power of each PC. Specifically, in the previous study,[21] by setting T_{cc} as the CGH computation time for each frame, the total CGH computation time for F frames simply becomes $T_{cc}F$. In contrast, in a simple case in which all the PCs have the same computing power, F/C frames are assigned to each PC, and the total CGH computation time in the proposed system is $T_{cp}F/C$. Although T_{cp} is considerably larger than T_{cc}, T_{cp}/C is equal to T_{cc} with a large C. This is also applicable when client PCs have different computing power because a smaller number of frames are assigned to the client with lower computing power. Consequently, the proposed system specializes in generating a video hologram with a large number of frames (at least $F \geq C$).

Notice that, in the proposed method where the parallelization is achieved on a frame-to-frame basis, each PC has a residual computational capacity as shown in Fig. 2. To resolve the problem, one can consider an approach that splits a frame into two (or more) subframes (i.e., distributing the light sources of a frame to different clients, which is similar to the previous study[21]) and assigns them to the residual space as shown in the lower figure of Fig. 2. However, since the CGH images (i.e., intermediate interference patterns) computed from the split frames have the same resolution as that of the CGH images (i.e., fully generated interference patterns) computed from the nonsplit frames, the data communication time is doubled. In turn, the benefit from minimizing the residual capacity by splitting the frame is larger than the loss associated with the increase in the data communication time.

2.2 *Parallelization of the Subprocesses through Multithreading*

By distributing the CGH computation on a frame-to-frame basis, the number of transmissions of the computation results from the client PCs to the server can be reduced. However, when the number of client PCs or the hologram resolution is high, the time taken for the reduced number of transmissions is still long. To resolve this problem, the proposed system executes the subprocesses (distribution, CGH computation, and collection) in parallel by multithreading. In other words, each client can get the light source information for the next frames or send the computation results for the previous frames to the server while performing the CGH computation for the current frame. With this scheme, if the time taken for both the distribution and collection subprocesses is shorter than that taken for the CGH computation subprocess (actually, this is very common), the total hologram generation time is fully determined by the CGH computation time and the data transmission time can be zero.

To make the subprocesses run in parallel, all the operations in each client PC and the server are implemented as thread functions that communicate with each other using the message passing method[22] (see Fig. 3). On the server side, the control thread decides how many frames to distribute to each client PC and the collect thread collects the computation results (i.e., fully generated interference pattern for each frame) from the client PCs and arranges them. On the client side, the compute thread computes the interference patterns for the assigned frames. The send and receive threads on both sides communicate the light source information of frames or the resulting interference patterns with each other. In each client PC, the receive thread sends a message to the compute thread after receiving the light source information from the server and then waits for the light source information for the next frame. The compute thread sends a message to the send thread after completing the CGH computation for the current frame and then waits for the message from the receive thread. The send thread sends the resulting interference pattern for the current frame to the server and then waits for the message from the computer thread. Consequently, the receive thread can receive the light source information for the next frames while the computer thread is performing the CGH computation for the current frame. The compute thread can perform the CGH computation for the next frames while the send thread is sending the interference pattern for the current frame to the server.

Notice that there is no memory problem occurred by parallelizing the subprocesses; thus, no elaborate memory management is required. In the distribution and computation subprocesses, the amount of light source data is very tiny and each frame is computed/generated sequentially (not in

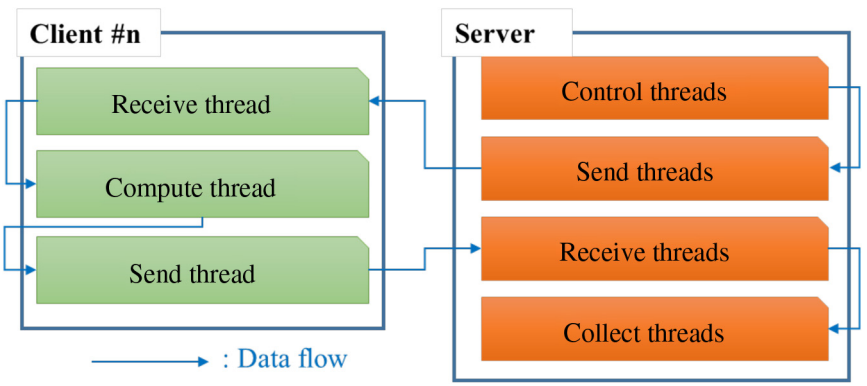

Fig. 3 Thread functions in each client PC and the server PC.

parallel) in the clients. This ensures that the clients need only a small amount of memory. In the collection subprocess, all the frames can arrive at the server at the same time in the worst case. However, this situation rarely happens, and the required memory amount is still not a big deal.

3 Experimental Results and Discussion

The performance of two proposed schemes, namely, frame distribution and multithreading (abbreviated to FD and MT hereafter), for reducing the data communication time in generating video holograms is evaluated.

3.1 *Effect of Changing the Way for Distributing CGH Computations*

A PC cluster was composed of six PCs (a server and five clients) that were connected to each other through a gigabit Ethernet hub (Cisco SG300-28[23]) and Winsock TCP/IP.[24] No network performance optimization was considered. Each client PC had one or two CUDA-enabled GPUs as shown in Table 1. In a manner similar to the previous study,[21] the "windmill" video was used as a 3-D object (see Fig. 4). The OpenCV[25] library and the CUDA API[26] were used for image processing and parallel processing, respectively.

Table 1 Specifications of each PC in the first PC cluster.

Role	GPU (quantity)	CPU	RAM (GB)	OS
Client_1	GeForce	i7	16	Windows
	GTX 980 (2)	4.0 GHz		8.1 Ent.
Client_2	GeForce	i7	32	Windows
	GTX 980 Ti (1)	4.0 GHz		8.1 Ent.
Client_3	GeForce	i7	8	Windows
	GTX 680 (1)	3.5 GHz		8.1 Pro
Client_4	GeForce	i7	32	Windows
	GTX TITAN (2)	3.6 GHz		8.1 Pro
Client_5	GeForce	i7	16	Windows
	GTX 580 (1)	3.5 GHz		8.1 Pro
Server	GeForce	i7	16	Windows
	GTX 750 Ti (1)	3.5 GHz		8.1 Pro

In Eq. (1), the reference wavelength was 532 nm and the pixel pitch was 8 μm.

Four experiments were performed to analyze how the hologram generation time varies in various conditions (number of client PCs, number of light sources, hologram resolution, and number of frames). Each experiment was repeated 10 times, and the results were averaged.

First, the CGH computations were performed for 100 frames with a hologram resolution of 2048×2048 and ~23,000 light sources. The computation times of two systems, namely, the proposed system (with FD only) and the system used in the previous study,[21] were measured while increasing the number of client PCs (see Table 2). The total time of the proposed system was continuously reduced but that of the previous system[21] was not. However, the core computation times (which corresponded to the difference between the total time and the data communication time) of both systems were similar and continuously decreased by increasing the number of client PCs. This was because the data communication time of the previous system[21] rapidly increased whereas that of the proposed system decreased. Consequently, the difference between the total computation times of both systems was potentially owing to the difference between their data communication times. With five clients, the proposed system assigned the frames of 30%, 21%, 8%, 39%, and 2% to each client in order and was ~2.1 times faster than that of the previous system.[21] The data communication time was ~10.0 times shorter.

Notice that the total computation time of the previous system[21] increased when $C > 3$. This is because the increase in data communication time was larger than the time saved by the distributed computation using multiple PCs. This indicates that the performance of the previous system[21] is strictly limited without resolving the increase in data communication time. Consequently, it is expected that the difference between the total computation times of the proposed system and the previous system[21] would be larger when $C > 5$.

Second, the CGH computations were performed for 100 frames with 1536×1536 hologram resolution and five client PCs. The computation times of the same two systems were measured while increasing the number of light sources (see Table 3). The data communication time was only slightly influenced by the number of light sources. With respect to the time for the system used in the previous study,[21] the total time gradually increased given that the study achieved the parallelization on a light source basis. However, the proposed system with the fixed number of frames was slowed

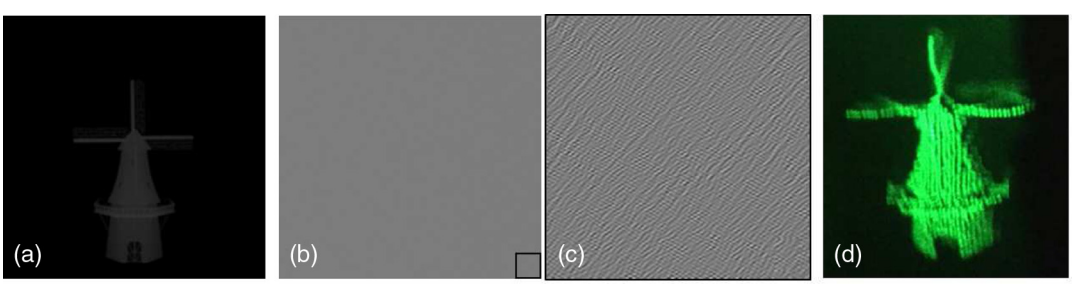

Fig. 4 3D object video used in our experiments. (a) 46th frame, (b) its CGH image (1536×1536), (c) enlargement of a region (black square) in the CGH image, and (d) the optical reconstruction image. The main purpose of this study is the rapid generation of the CGH image for each frame.

Table 2 CGH computation time (ms) per frame according to the number of cluster PCs.

	Total time			Data communication time		
C	Previous[21]	Proposed (FD)	Ratio	Previous[21]	Proposed (FD)	Ratio
1	1166 (969)*	1165 (971)	1.0	197	194	1.0
2	745 (526)	638 (530)	1.2	219	108	2.0
3	686 (402)	486 (404)	1.4	284	82	3.5
4	706 (389)	431 (370)	1.6	317	61	5.2
5	777 (363)	372 (332)	2.1	414	40	10.0

*The value within parentheses represents the core computation time except the data communication time.

Table 3 CGH computation time (ms) per frame according to the number of light sources.

	Total time			Data communication time		
L	Previous[21]	Proposed (FD)	Ratio	Previous[21]	Proposed (FD)	Ratio
≈1600	362	66	5.5	346	45	7.7
≈5300	380	79	4.8	330	36	9.2
≈13,000	413	135	3.1	296	31	9.5
≈23,000	466	242	1.9	220	31	7.1
≈35,000	508	338	1.5	197	25	7.9

down in proportion to the number of light sources. Consequently, the ratio between the total time of both systems continuously decreased owing to the increase in the number of light sources. This implies that the proposed system may not be suitable for the case with a small number of frames and a huge number of light sources. In our experiments, although the total time of the proposed system was still shorter than that of the previous system,[21] this can be reversed with more than 60,000 of the light sources as shown in Fig. 5.

Third, the CGH computations were performed for 100 frames with ∼23,000 light sources and five client PCs. The computation times of the same two systems were measured while increasing the hologram resolution (see Table 4). The core computation times of both systems were similar and

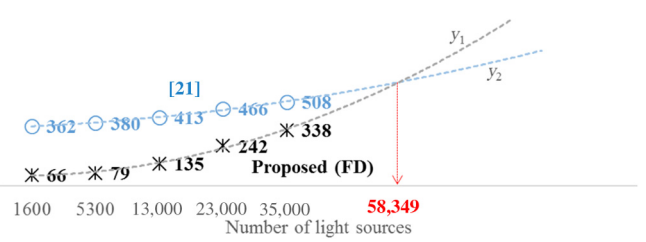

Fig. 5 Plotting the total computation time in Table 3 and its second-order polynomial extrapolation. $y_1 \approx 42.04814 + 0.007531116x + 2.921986e-8x^2$ and $y_2 \approx 352.3976 + 0.005296294x - 2.332514e-8x^2$.

equally increased by increasing the hologram resolution. However, the data communication time for the previous system[21] indicated a significantly rapid increase (this was because the difference between $CT_t(C=5)$ and T_t increased as T_t increased); thus, the ratio between the total time of both systems was maintained as ≈2.

Fourth, the CGH computations were performed with 1536×1536 hologram resolution, ∼23,000 light sources, and five client PCs. The computation times of three systems, namely, the proposed system, a system that parallelized the CGH computation on a frame-to-frame basis (same as the proposed system) but assigned the same number of frames to each client PC, and the system in the previous study[21] were measured while increasing the number of frames (see Table 5). The results indicated that the previous system[21] was faster than the other systems for the single frame case. However, since its data communication time was lengthy when compared with that of the other systems (the ratio between the data communication time of the previous system[21] and those of the other systems increased as the number of frames increased), the previous system[21] was slower than the other systems for the cases with two frames and higher. The difference in the total times of the previous system[21] and the proposed system increased as the number of frames increased. In Fig. 6, while the core computation time of the previous system[21] was almost constant with respect to the number of frames, the core computation time of both the uniform-distribution and the proposed systems decreased as the number of frames increased. With more than 30 frames, the core computation time of the proposed system became similar to or slightly shorter than that of the previous system[21] (as mentioned before, the efficiency of the proposed system comes from reduction in the data communication time). Note that the core computation time of the uniform-distribution system could not be reduced below 400 ms. This led to the difference between the total time of the proposed adaptive-distribution system and the uniform-distribution system. In the experiments where each frame had the same number of light sources, the total time of the proposed system was saturated at 100 frames or higher. The proposed system could be more advantageous if each frame had different numbers of light sources. [The uniform-distribution system distributes the frames to clients evenly, regardless of what numbers of light sources each frame has. This has a risk of distributing the frames that have a number of light sources to a low-performance client PC. In contrast, the proposed

Table 4 CGH computation time (ms) per frame according to the hologram resolution.

	Total time			Data communication time		
$W_h \times H_h$	Previous[21]	Proposed (FD)	Ratio	Previous[21]	Proposed (FD)	Ratio
512^2	58 (25)*	29 (23)	2.0	33	6	5.5
1024^2	220 (94)	111 (93)	2.0	126	18	7
1536^2	466 (246)	242 (211)	1.9	220	31	7.1
2048^2	777 (363)	372 (332)	2.1	414	40	10.0
2560^2	1182 (569)	621 (543)	1.9	613	78	7.9

*The value within parentheses represents the core computation time except the data communication time.

Table 5 CGH computation time (ms) per frame according to the number of frames [Ψ in Eq. (3)].

	Total time			Data communication time		
F	Previous[21]	Uniform*	Proposed (FD)	Previous[21]	Uniform*	Proposed (FD)
1	592	776	776	323	99	99
2	612	468	491	359	45	72
5	524	445	448	270	23	41
10	527	442	345	273	22	51
30	465	442	310	216	28	51
50	456	437	288	211	32	38
70	461	432	269	215	26	40
100	466	431	242	220	26	31
140	461	429	245	215	35	31
180	462	429	244	216	25	30

*The system that uniformly assigns the number of frames to each client PC.

system can readily handle this problem, by modifying Eq. (3) to adaptively distribute the frames while taking into consideration the number of light sources that each client PC has.]

Table 6 Specifications of each PC in the second PC cluster.

Role	GPU (quantity)	CPU	RAM	OS
Client_1	GeForce	i7	16 GB	Windows
	GTX 980 (2)	4.0 GHz		8.1 Ent.
Client_2	GeForce	i7	32 GB	Windows
	GTX 980 Ti (1)	4.0 GHz		10 Edu.
Client_3	GeForce	i7	8 GB	Windows
	GTX 680 (1)	3.5 GHz		8.1 Pro
Client_4	GeForce	i7	32 GB	Windows
	GTX 580 (2)	3.6 GHz		8.1 Pro
Server	GeForce	i7	32 GB	Windows
	GTX 770 (1)	3.4 GHz		8.1 Pro

3.2 *Effect of Using Multithreading*

A slightly different PC cluster was used (see Table 6), but the other experimental environments were almost the same as the previous experiments.

First, 150 frames were generated with a hologram resolution of 1024×1024 and ~23,000 light sources. The generation times of two systems, namely, the proposed system (with FD only) and the proposed system (with both FD and MT), were measured while increasing the number of client

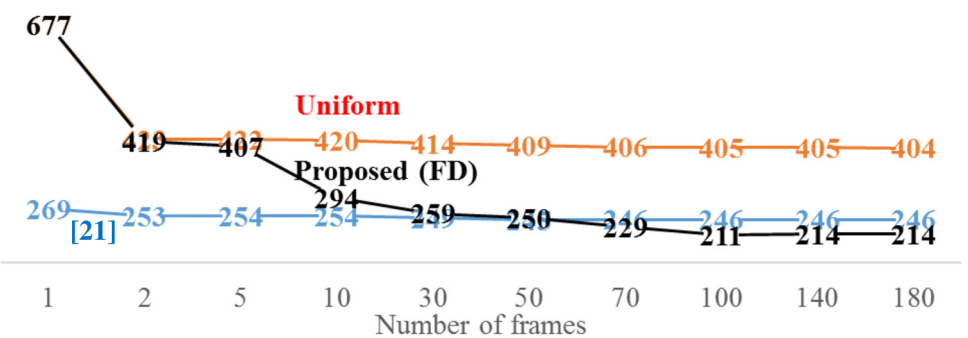

Fig. 6 Core computation time (ms) in Table 5.

Table 7 Video hologram generation time (s) according to the number of cluster PCs.

					Total generation time		
C		Ψ_c	T_{cp}	T_t	Proposed (FD)	Proposed (FD + MT)	Ratio
2	Client_3	64	0.647	0.026	121.746	120.861	1.01
	Client_4	86					
3	Client_2	80	0.267	0.022	58.452	51.782	1.12
	Client_3	29					
	Client_4	41					
4	Client_1	66	0.147	0.023	33.466	28.332	1.18
	Client_2	44					
	Client_3	17					
	Client_4	23					

PCs (see Table 7). The separate CGH computation time and data transmission time of both systems were similar, and the total generation times of both systems were continuously decreased. However, by using multithreading, the data communication time could be further reduced (because the collection subprocess are running in the background) and the proposed system with both schemes was faster by increasing the number of client PCs. With four client PCs, the proposed system with both FD and MT was ~1.2 times faster than with FD only.

Second, 150 frames were generated with 1024×1024 hologram resolution and four client PCs. The generation times of the same two systems were measured while increasing the number of light sources (see Table 8). As expected, both systems with the fixed number of frames were slowed down in proportion to the number of light sources. In particular, the CGH computation time was much longer than the data transmission time. This gradually reduced the benefit from using multithreading. Consequently, although the total generation time of the proposed system could always be shorter by using multithreading, the speedup index of 1.56 with 5300 light sources was decreased to 1.10 with 50,000 light sources.

Third, 150 frames were generated with ~23,000 light sources and four client PCs. The generation times of the same two systems were measured while increasing the hologram resolution (see Table 9). As already observed in Table 4, both the CGH computation time and the data communication time increased together and at the same rate by increasing the hologram resolution. Consequently, regardless of the hologram resolution, the proposed system with both FD and MT was ~1.17 times faster than with FD only.

Fourth, in the experiment of Table 8, the MT scheme was applied to the previous system.[21] For each frame, the three processes for distribution of light sources, CGH computation on the client side, and collection of the partial interference patterns from the clients were parallelized through multithreading. As shown in Table 10, the previous system was also greatly improved although the improvement was gradually lost in proportion to the number of light sources. This indicates that the MT scheme is useful for the previous system as well. Actually, the MT scheme was more effective for the previous system because of the higher percentage of the data transmission time. Compared with the results of Table 8, the previous system with MT could be faster than the proposed system with FD only when using a large number of light sources. This presents the impact of the MT scheme. However, with a small number of light sources, the proposed system with FD only was faster. The more important thing is that the proposed system with both FD and MT was always faster (maximally 4.3 times faster) than the previous systems

Table 8 Video hologram generation time (s) according to the number of light sources.

			Total generation time		
L	T_{cp}	T_t	Proposed (FD)	Proposed (FD + MT)	Ratio
≈5300	0.033	0.035	13.155	8.391	1.56
≈13,000	0.078	0.034	19.654	15.424	1.28
≈23,000	0.147	0.023	33.466	28.332	1.18
≈50,000	0.326	0.019	69.008	62.830	1.10

Table 9 Video hologram generation time (s) according to the hologram resolution.

			Total generation time		
$W_h \times H_h$	T_{cp}	T_t	Proposed (FD)	Proposed (FD + MT)	Ratio
512^2	0.035	0.005	6.979	6.039	1.16
1024^2	0.147	0.023	33.466	28.332	1.18
1536^2	0.363	0.048	74.318	64.143	1.16

Table 10 Video hologram generation time (s) of the previous study[21] according to the number of light sources.

	Total generation time		
L	Previous[21]	Previous[21]+MT	Ratio
≈5300	36.231	22.900	1.59
≈13,000	37.961	23.683	1.61
≈23,000	39.989	28.957	1.39
≈50,000	69.803	63.737	1.10

Table 11 Video hologram generation time (s) of the previous study[21] according to the hologram resolution.

	Total generation time		
$W_h \times H_h$	Previous[21]	Previous[21] + MT	Ratio
512^2	12.743	8.579	1.49
1024^2	39.989	28.957	1.39
1536^2	98.079	66.973	1.47

with and without MT. Therefore, we can safely say that both the schemes FD and MT are necessary for fast generation of a video hologram.

Finally, in the experiment of Table 9, the MT scheme was applied to the previous system.[21] As shown in Table 11, the improvement by MT was significant and consistent regardless of the hologram resolution. When the hologram resolution was high, the previous system with MT could be faster than the proposed system with FD only (see the results for $L \approx 50{,}000$ in Tables 9 and 11). However, the proposed system with both FD and MT was always faster (maximally 2.1 times faster) than the previous systems with and without MT. Therefore, it is clear again that both the schemes FD and MT are necessary for fast generation of a video hologram.

4 Conclusion

This study proposed a PC cluster system that efficiently generated a video hologram. The system first parallelized the hologram generation on a frame-to-frame basis to reduce the data communication time between client PCs and the server and thus specialized in generating a video hologram with a large number of frames. In addition, the system could optimally distribute the number of computations to each PC according to its computing power. The efficiency of the proposed system was evident in the experiment. For a video hologram with 100 frames, 1536×1536 hologram resolution, and ∼23,000 light sources, the proposed system (composed of five client PCs) generated each frame in 242 ms. This was 1.9 times shorter than the system that parallelized the computations for generating each individual frame and 1.8 times shorter than the system that equally distributed the number of computations to each PC.

Then, the proposed system also enabled the subprocesses for generating each frame of a video hologram to execute in parallel through multithreading. This made the data communication time close to zero and thus enabled the proposed system (composed of four client PCs) to be additionally 1.2 times faster in the experiment where a video hologram with 150 frames and ∼23,000 light sources was generated.

With the proposed schemes for reducing the data communication time, it could be expected that the hologram generation time would be further reduced by increasing the number of client PCs. Therefore, it would be interesting to analyze the performance of the proposed system with many more PCs. In addition, the performance of the proposed system will depend on the other system configurations (specifications or topology of client PCs). Therefore, in the near future, we are going to explore how to set up a set of clusters that is more optimal.

Acknowledgments

This research was supported by the Ministry of Culture, Sports and Tourism and Korea Creative Content Agency in the Culture Technology Research and Development Program 2017.

References

1. D. Gabor, "A new microscopic principle," *Nature* **161**, 777–778 (1948).
2. B. R. Brown and A. W. Lohmann, "Complex spatial filtering with binary masks," *Appl. Opt.* **5**, 967–969 (1966).
3. H. Yoshikawa and J. Tamai, "Holographic image compression by motion picture coding," *Proc. SPIE* **2652**, 2–9 (1997).
4. K. Matsushima, H. Schimmel, and F. Wyrowski, "Fast calculation method for optical diffraction on tilted planes by use of the angular spectrum of plane waves," *J. Opt. Soc. Am. A* **20**(9), 1755–1762 (2003).
5. P. Su et al., "Fast computer-generated hologram generation method for three-dimensional point cloud model," *J. Disp. Technol.* **12**(12), 1688–1694 (2016).
6. Y. Ogihara and Y. Sakamoto, "Fast calculation method of a CGH for a patch model using a point-based method," *Appl. Opt.* **54**(1), A76–A83 (2015).
7. M. Lucente, "Interactive computation of holograms using a look-up table," *J. Electron. Imaging* **2**, 28–34 (1993).
8. S. C. Kim and E. S. Kim, "Effective generation of digital holograms of three-dimensional objects using a novel look-up table method," *Appl. Opt.* **47**(19), D55–D62 (2008).
9. H. Yoshikawa, "Fast computation of Fresnel holograms employing difference," *Opt. Rev.* **8**(5), 331–335 (2001).
10. T. Shimobaba and T. Ito, "An efficient computational method suitable for hardware of computer-generated hologram with phase computation by addition," *Comput. Phys. Commun.* **138**(1), 44–52 (2001).
11. T. Nishitsuji et al., "Simple and fast cosine approximation method for computer-generated hologram calculation," *Opt. Express* **23**(25), 32465–32470 (2015).
12. Z. Chen et al., "Acceleration for computer-generated hologram in head-mounted display with effective diffraction area recording method for eyes," *Chin. Opt. Lett.* **14**(8), 080901 (2016).
13. Y. Ichihashi et al., "HORN-6 special-purpose clustered computing system for electroholography," *Opt. Express* **17**(16), 13895–13903 (2009).
14. N. Masuda et al., "Computer generated holography using a graphics processing unit," *Opt. Express* **14**(2), 603–608 (2006).
15. T. Shimobaba et al., "Fast calculation of computer-generated-hologram on AMD HD5000 series GPU and OpenCL," *Opt. Express* **18**, 9955–9960 (2010).
16. Y. Pan, X. Xu, and X. Liang, "Fast distributed large-pixel-count hologram computation using a GPU cluster," *Appl. Opt.* **52**, 6562–6571 (2013).
17. J. Song et al., "Real-time generation of high-definition resolution digital holograms by using multiple graphic processing units," *Opt. Eng.* **52**(1), 015803 (2013).
18. T. Sugawara, Y. Ogihara, and Y. Sakamoto, "Fast point-based method of a computer-generated hologram for a triangle-patch model by using a graphics processing unit," *Appl. Opt.* **55**, A160–A166 (2016).
19. N. Takada et al., "Fast high-resolution computer-generated hologram computation using multiple graphics processing unit cluster system," *Appl. Opt.* **51**(30), 7303–7307 (2012).
20. H. Niwase et al., "Real-time electroholography using a multiple-graphics processing unit cluster system with a single spatial light modulator and the InfiniBand network," *Opt. Eng.* **55**(9), 093108 (2016).

21. J. Song et al., "Fast generation of a high-quality computer-generated hologram using a scalable and flexible PC cluster," *Appl. Opt.* **55**(13), 3681–3688 (2016).
22. "Messages and message queues," https://msdn.microsoft.com/en-us/library/windows/desktop/ms632590(v=vs.85) (18 November 2017).
23. "Cisco gigabit managed switch," http://www.cisco.com/c/en/us/support/switches/sg300-28-28-port-gigabit-managed-switch/model.html (18 November 2017).
24. K. R. Fall and W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, 2nd ed., Addison-Wesley Professional, Boston, Massachusetts (2011).
25. "OpenCV," http://opencv.org/ (18 November 2017).
26. "CUDA zone," https://developer.nvidia.com/cuda-zone (18 November 2017).

Hanhoon Park received his BS and MS degrees and PhD in electrical and computer engineering from Hanyang University, Seoul, Korea, in 2000, 2002, and 2007, respectively. From 2008 to 2011, he was a postdoctoral researcher with NHK Science and Technology Research Laboratories, Tokyo, Japan. In 2012, he joined the Department of Electronic Engineering, Pukyong National University, Busan, Korea, where he is currently an associate professor. His current research interests include augmented reality, human–computer interaction, and affective computing.

Joongseok Song received his BS degree in electronics engineering from Korea Polytechnic University, Siheung, Korea, in 2010. He received his MS degree and PhD in electronics and computer engineering from Hanyang University, Seoul, Korea, in 2012 and 2016, respectively. He is currently a researcher in the Software Laboratory, Kohyoung Technology, Seoul, Korea. His research interests include image processing and 3-D computer vision, digital holography, and GPGPU.

Changseob Kim received his BS degree in computer science from Hanyang University, Seoul, Korea, in 2017, and he is currently working toward his MS degree. His research interests include hologram, GPGPU, 3-D computer vision, augmented reality, and tracking.

Jong-Il Park received his BS and MS degrees and PhD in electronics engineering from Seoul National University, Seoul, Korea, in 1987, 1989, and 1995, respectively. From 1996 to 1999, he was a researcher with the ATR Media Integration and Communication Research Laboratories, Kyoto, Japan. In 1999, he joined the Department of Electrical and Computer Engineering at Hanyang University, Seoul, Korea, where he is currently a professor. His research interests include computational imaging, augmented reality, 3-D computer vision, and human–computer interaction.