Optical Engineering

SPIEDigitalLibrary.org/oe

Consideration of time to create real-time video of a kinoform using a web camera

Masataka Tozuka
Kunihiko Takano
Makoto Ohki
Koki Sato

SPIE

Consideration of time to create real-time video of a kinoform using a web camera

Masataka Tozuka
Shonan Institute of Technology
Fujisawa, 251-8551 Japan
E-mail: tozu3m1@hope.nifty.jp

Kunihiko Takano
Tokyo Metropolitan College of Industrial Technology
Tokyo, 116-8523 Japan

Makoto Ohki
Koki Sato
Shonan Institute of Technology
Fujisawa, 251-8551 Japan

Abstract. Using a two-dimensional fast Fourier transform is an efficient way to calculate a kinoform. High-speed processing of large amounts of data points (e.g., a 512×512 matrix) can be accomplished using a kinoform. Real-time computer-generated hologram calculation has been widely pursued. To this end, use of the graphics processing unit (GPU) or multiprocessing methods are becoming popular for high-speed processing. We used the GPU method coupled with multiprocessing to construct a kinoform and measured the efficiency of this method. *© 2011 Society of Photo-Optical Instrumentation Engineers (SPIE).* [DOI: 10.1117/1.3596384]

Subject terms: kinoform, fast Fourier transform; real-time; graphics processing unit; compute unified device architecture.

Paper 101037SSR received Dec. 15, 2010; revised manuscript received Apr. 6, 2011; accepted for publication Apr. 22, 2011; published online Aug. 3, 2011.

1 Introduction

We aimed to develop a system that uses a single personal computer to create a real-time interference fringe. This system makes the real-time animation of a real three-dimensional (3D) object with an inexpensive web camera through use of a simple kinoform-type pattern. Approximately 10 frames of the 30 frames/s recorded by the web camera were used to create the kinoform of the real-time animation that a kinoform-type pattern is computed for each frame of a movie. A kinoform allows for the high-speed processing of a significant amount of data (e.g., 262,144 elements in a 512×512 matrix); the fact that this processing methodology allows for the reconstruction of an image using neutral colors is unique because it enables the computation of an interference fringe at very high speeds by calculating the two-dimensional fast Fourier transform (2D-FFT) with a personal computer.[1] Other methods used to increase the calculation speed with which a computer-generated hologram (CGH) is able to create a table have been reviewed previously.[2] Recently, there has been increasing interest in the use of a graphics processing unit (GPU) or a custom large scale integration[3–8] to perform general computations, such as an FFT, due to the higher throughput capacity of these devices. Current methods do not utilize the increased speed of the GPU for the general computation of the 2D-FFT using a kinoform, resulting in the need for multiple calculations. The method described here compares the kinoform creation processing time when using a CPU versus the time used by a GPU. In addition, we review the features that are promising for real-time processing that allows for the recording of an animation of a real 3D object using an inexpensive web camera.

2 Kinoform

Our method uses an FFT when synthesizing a CGH. The real part (R_{xy}) of the complex amplitude distribution (T_{xy}) of the computed wavefront (which should record an imaginary part, I_{xy}) is indicated in Fig. 1. The wavefront amplitude A_{xy} and phase φ_{xy} are given by formulas (1) and (2) below.

Amplitude

$$\mathrm{A_{xy}} = \sqrt{\mathrm{R_{xy}^2} + \mathrm{I_{xy}^2}}, \tag{1}$$

*$\mathrm{A_{xy}}$ of Formula (1) is about constant.

$$\text{Phase } \varphi_{\mathrm{xy}} = \tan^{-1}\frac{\mathrm{I_{xy}}}{\mathrm{R_{xy}}}. \tag{2}$$

These formulas assume that the wavefront amplitude A_{xy} is constant, which is caused by the random noise, and it reconstructs an image only from the information provided by the phase (φ_{xy}). The phase value (φ_{xy}) is distributed on the kinoform surface in 256 levels; this phase modulation is imparted to the liquid crystals using a voltage change (Fig. 2). In addition, optical phase modulation is also possible.

3 Methods

Our method depends on the kinoform, which uses a method to create the CGH for the optical reconstruction that relies on a twisted-nematic (TN)-type LCD. As indicated in Fig. 3, the process begins by recording an animation of the real 3D object with a web camera. The animation is converted to black and white, then begins to finish cutting off a 512×256 portion

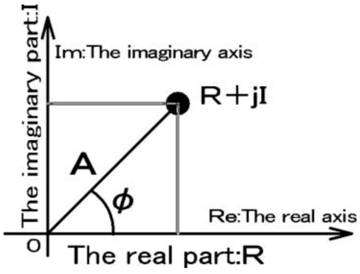


Fig. 1 The complex plane.

0091-3286/2011/$25.00 © 2011 SPIE

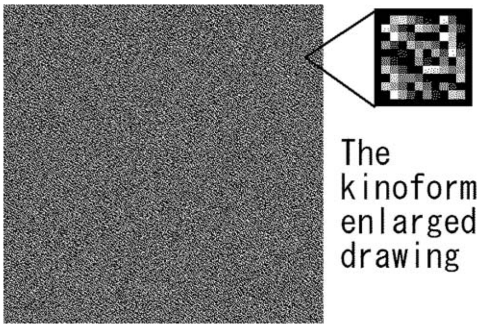


Fig. 2 A 512×512 kinoform pattern.

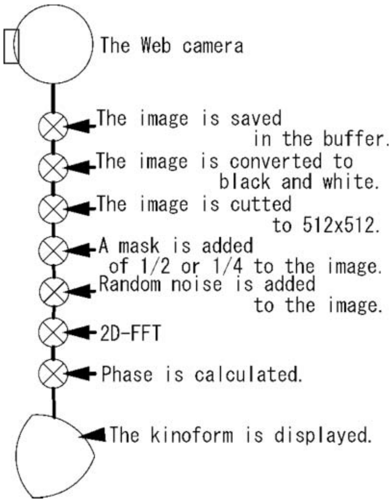

The Web camera
The image is saved in the buffer.
The image is converted to black and white.
The image is cutted to 512x512.
A mask is added of 1/2 or 1/4 to the image.
Random noise is added to the image.
2D-FFT
Phase is calculated.
The kinoform is displayed.

Fig. 3 The kinoform creation process.

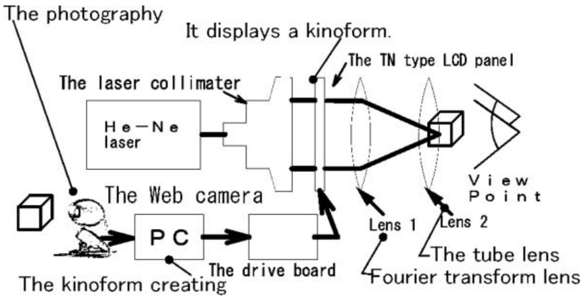


Fig. 4 The kinoform reconstruction.

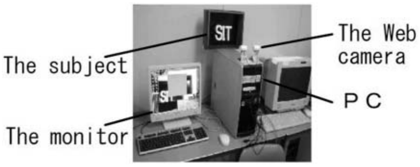


Fig. 5 The kinoform creation and photography environment.

Table 1 The variable distribution used in the FFT to create the 512×512-element kinoform.

It	0	. . .	6	7	8
xp^2	256	. . .	4	2	1
k_0	0	. . .	0	0	0
k_1	1	. . .	1	1	—
k_2	2	. . .	2	—	—
⋮	⋮	. . .	3	—	—
k_{255}	255	. . .	—	—	—

of the original image to create the 512×512 kinoform with 2D-FFT. It avoids the issue in which the real images and conjugate images (composing the reconstructed image) begin to pile up; it shifts the image to the 512×512 square half in the CPU and GPU methods. The GPU was programmed using compute unified device architecture (CUDA), which is provided by NVIDIA, Inc..[3,4] The GPU used for this study was the GeForce 8800 GTX, which has 128 integration shaders that operate in parallel and allows for 345.6-GFLOPS single-precision operation at peak performance.

Method (a) When using CUDA to create the kinoform from the 2D-FFT (if using the CUFFT library that depends on CUDA), the FFT can be simply and quickly executed on the GPU. The GPU uses the CUFFT library function cufftExecC2C to compute the FFT. Therefore, it is not necessary to tabulate sine or cosine values to perform the 2D-FFT with CUDA. Method (c) or (d) of after relies on a CPU to process the table of the random noise (e.g., a 512×512 matrix) (or function into the direct of random noise processing) in addition to the GPU processing.

Table 2 Specifications of the personal computer.

Quantity	One	
Memory	2.00 GB RAM	
CPU	Intel® Core^{TM} 2 Duo processor E6850 (3 GHz)	
GPU	NVIDIA GeForce 8800 GTX	
	Core clock	576 MHz
	Memory clock	900 MHz
	Dater rate	1800 MHz
	Memory	768 MB
	Stream-processor	128 units
OS	Windows® XP Home Edition Version 2002 SP3	

Table 3 Specifications of the web camera.

Quantity	One
I/F	USB connection
Photo acceptance unit	CMOS sensor (1/6 inch)
Maximum resolution	640×480 pixels
Frame rate	30 frames p/s
Number of colors	16,770,000 colors (24-bit)

Method (b) As indicated in Table 1, along with formulas (3) and (4), the sine or cosine values are computed beforehand and then arranged to be used by the CPU during the FFT calculation to compute the CGH, and Method (c) or (d) rely on a CPU to process the table of the random noise (e.g., a 512×512 matrix) (or function into the direct of random noise processing).

$$\langle \text{Realpart} \rangle$$
$$T\ \cos[\text{it}][k] = \cos(k \times \pi / xp^2), \tag{3}$$
$$\langle \text{Imaginarypart} \rangle$$
$$T\ \sin[\text{it}][k] = \sin(k \times \pi / xp^2), \tag{4}$$

Method (c) A method of making the table the addition to the GPU method (a) and the CPU method (b), this method adds, from the Rand_{xy} matrix described in formula (5), a random amount of noise into the 512×512 matrix, which is then used to create the kinoform in advance and arrange it such that it can be used to compute the CGH.

$$\text{Rand}_{xy} = [\text{randomnumbers}], \tag{5}$$

Method (d) A method of making the table the addition of the random noise given in method (c) is expanded in method (d). In addition to the GPU method (a) and using only the CPU method (b), this method computes the $\cos(2\pi \times \text{random number})_{xy}$ matrix used in Eq. (6) and the $\sin(2\pi \times \text{random number})_{xy}$ matrix used in Eq. (7) to add to the input in advance, and then arrange it to be used to compute the CGH. Here, b_{xy} refers to the image data collected by recording an animation.

$$\langle \text{Realpart} \rangle$$
$$\text{Are}_{xy} = b_{xy} \times \exp(0.0) \times \cos(2\pi \times \text{Rand}_{xy})_{xy}, \tag{6}$$
$$\langle \text{Imaginarypart} \rangle$$
$$\text{Aim}_{xy} = b_{xy} \times \exp(0.0) \times \sin(2\pi \times \text{Rand}_{xy})_{xy}. \tag{7}$$

Table 4 Specifications of the TN-type LCD.

Quantity	One
screen size	15.408(H)×8.688(V) (mm)
dot pitch	12(H)×12(V) (um)
number of dots	1284(H)×742(V)
numerical aperture	48%

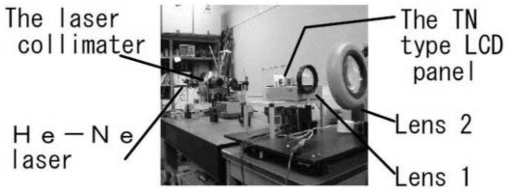


Fig. 6 The kinoform reconstruction optical system environment.

4 Results

As indicated in Figs. 4 and 5, the program was created using Visual C++ (Visual Studio 2008 Professional Edition C++) and CUDA (NVIDIA CUDA, version 2.1) with a single personal computer (Table 2) and one web camera (Table 3). These figures illustrate the creation of the kinoform presented in Fig. 2. As indicated in Figs. 4 and 6 using a TN-type LCD (Table 4), the system uses a He–Ne laser (632-nm wavelength) and a laser collimator acquired from Neo Arc Co. Ltd. This system performs a Fourier transformation of the kinoform with lens 1, and the image formation can be visualized with lens 2. The reconstructed image can be observed with two eyes, as illustrated in Fig. 7.

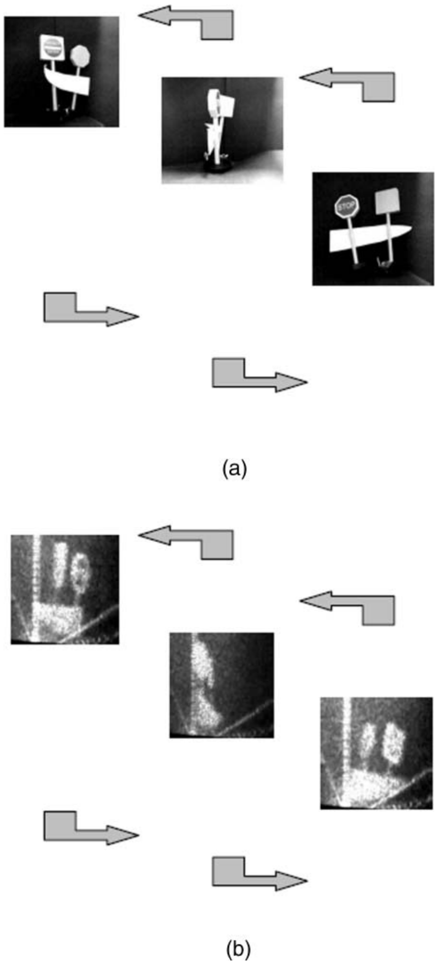
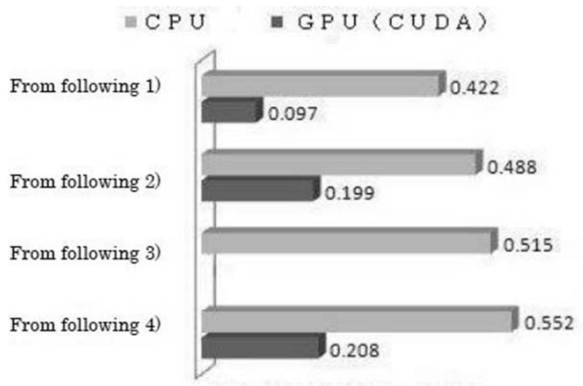


Fig. 7 The real-time kinoform reconstructed image: (a) the real 3D object image of the camera captures the video (the real 3D object is a sign), and (b) a video of the reconstructed image. (QuickTime, 1.1 MB) [URL: http://dx.doi.org/10.1117/1.3596384.1]



1) It uses a table to addition processing in general of the random number.

2) It uses a table for the random number.

3) It uses a table for sin/cos of the Fourier transformation.

4) Only the basic function

Fig. 8 The measurement process with operation time given.

The black and white pictures shown in Fig. 7 recorded the animation at 10.3 fps.

5 Discussion

The cycle time required by the CPU and GPU, the images recorded by the web camera, the 2D-FFT processing, and the kinoform are all diagrammed in Fig. 9; these were measured using the C++ time library. The cycle time required for each step is indicated in Fig. 8.

The results indicated that the operation times increase in the following order: GPU (table) < GPU < CPU (table) < CPU (C++). According to these results, the GPU requires less than or equal to 1/4 the operation time of the CPU. Therefore, use of a GPU is ideal for rapid kinoform creation.

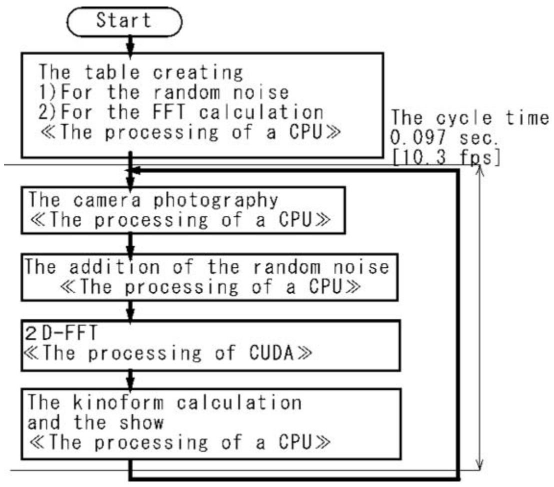


Fig. 9 Comparison of the cycle time (unit/s)

Because web camera imaging is buffered during animation, the operation time depends mostly on the 2D-FFT and a few other influences. This was independent of the processing that was required for the kinoform created by the 2D-FFT. The operation time increases by about 0.01 s due to the photography of the web camera if the operation time includes the showing of the image.

A big portion of the time load arises from the change of the operation time of Fig. 8 to the CPU during the time required to begin addition of the data of the random number by becoming a table. As indicated in Fig. 9, there was a large difference between the results in the cases of use of the CPU only and the cases with the addition of the GPU.

The result of calculation of only the 2D FFT by the GPU yielded an interesting result. It was observed that the majority of the increase in speed of the kinoform creation process was due to an increase in 2D-FFT speed.

6 Conclusion

The superior performance of the GPU relative to the CPU resulted in a more rapid computation. The real-time processing of images of a real 3D object taken by a web camera using a kinoform-type CGH processing methodology is a technique that can be sped up using general purpose GPU technology. Using a GPU also meant using CUDA, which was provided by NVIDIA, Inc. The source code for C was developed using Microsoft Corporation's Visual Studio in the Windows environment. Future research should be able to leverage this work.

In future work, the use of multiple GPUs will be explored to further speed up real-time kinoform processing.

Acknowledgments

The authors wish to express their gratitude to Dr. K. Takano, Dr. M. Ohki, and Dr. K. Sato for their assistance in providing guidance with regard to conference presentations and research in general. In addition, the authors are grateful to all members of the laboratory of Dr. K. Sato who cooperated in this study.

References

1. K. Sato, M. Tozuka, K. Kiuchi, "Characteristics of Kinoform by plural LCD panels and its application to display the color image," *J. Inst. Image Inf. Telev. Eng*. **29**, 331–332 (1993).
2. T. Yatagai, "The spread of the computer-generated hologram technology," http://optlab2.bk.tsukuba.ac.jp/jp/people/yatagai/Lectures/WWW-CGH.htm (2009).
3. T. Shimobaba and T. Ito, "Applying the parallel processing technology sophisticated in graphics processing field to general purpose numerical computing: The fact of GPU (graphics processing unit) computing using the CUDA technology," CQ Publishing Co., Ltd., Interface (2008).
4. T. Shimobaba and T. Ito, "High performance computation of optical wave propagation (Fresnel diffraction) using FFT on GPU: The fact of GPU (graphics processing unit) computing using the CUDA technology (2)," CQ Publishing Co., Ltd., Interface (2008).
5. T. Mizukami, Y. Ichihashi, T. Shimobaba, H. Nakayama, A. Shiraki, N. Masuda, T. Ito, "One-Unit System for Electroholography Using a Special-Purpose Computer with a High-Resolution Liquid Crystal Display," *J. Inst. Image Inf. Telev. Eng*. **62**(11), 1874–1876 (2008).
6. O. Nishikawa, T. Okada, K. Matsumoto, T. Ito, M. Taiji, H. Yoshikawa, and T. Honda, "Exclusive LSI For Calculation of Fresnel Hologram," *J. Inst. Image Inf. Telev. Eng*. **32**, 96–97 (1996).
7. S. Matsuda, T. Fujii, T. Yamaguchi, and H. Yoshikawa, " Fast Generation of Computer-Generated Disc Hologram by Graphics Processing Unit," *IEICE* **108**(425), 223–227 (2009).
8. H. Yoshikawa, "Computational speed of Fresnel holograms," *IEICE*, 241 (1994).

Masataka Tozuka graduated from the department of engineering at the Shonan Institute of Technology, Japan, in 1993. He joined Ando Electric Co. Ltd. in the same year. He is currently employed by the Yokogawa Manufacturing Co. Ltd.

Kunihiko Takano received his PhD in electrical engineering from the Shonan Institute of Technology, Japan, in 2002. He is now studying electro-holographic displays and remote medicine. Since April 2002, he has worked for the Tokyo Metropolitan College of Industrial Technology.

Makoto Ohki graduated from Junior College of Technology, Gunma University and received his PhD degree in electronic engineering from Gunma University in 1973 and 2000, respectively. From 1969 to 2001 he was an assistant engineer at the department of electronic engineering, Gunma University. Since 2001 he has been a lecturer of Electrical and Electronic Engineering, Shonan Institute of Technology. He is now an associate professor of electrical and electronic engineering. His interests are electromagnetic wave propagation, scattering and diffraction. He is a member of IEICE, IIEEJ, SPIE, and IEEE.

Koki Sato received his BE degree in electronics and communication engineering in addition to ME and PhD degrees in electrical engineering from Waseda University, Tokyo Japan, in 1970, 1972, and 1975, respectively. In 1977 he joined the electrical department of the Sagami Institute of Technology, Fujisawa, Japan. He is now a professor of the applied computer science department of the Shonan Institute of Technology (the school name changed in 1990). His research interests are optoelectronics and holography. He is a member of the Institute of Image Information and Television Engineers (fellow) and SPIE.