Recurrence formulas for fast creation of synthetic three-dimensional holograms

Kyoji Matsushima and Masahiro Takai

A method for accelerating the synthesis of computer-generated three-dimensional (3-D) holograms, based on conventional ray tracing, is proposed. In ray tracing, computers expend almost all of their resources in calculating the 3-D distances between each one of the point sources composing an object and a sampling point on the hologram. We present recurrence formulas that precisely compute the distances and reduce the computation time for synthesizing holograms to one half to one quarter, depending on the processor type. We demonstrate that a full-parallax hologram with an area of 4800 × 4800 pixels, synthesized for a 3-D object containing 966 point sources of light, is computed within 17 min and is optically reconstructed. © 2000 Optical Society of America

OCIS codes: 000.4430, 090.1760, 090.2870.

1. Introduction

Computer-generated holograms or digital holograms are useful for creating three-dimensional (3-D) images of virtual objects.[1,2] Recent electron-beam lithography techniques enable us to fabricate synthetic holograms with submicrometer structures.[3] In such large-scale holograms, however, a computational explosion in the numerical calculation of complex amplitude distributions for 3-D objects causes a bottleneck. Most of the computer-generated holograms described in the literature use the discrete Fourier transform of two-dimensional (2-D) image planes.[4–7] In this type of computer-generated hologram the required computation capacity is not so large, since the fast-Fourier-transform algorithm is applicable to hologram synthesis. However, it is difficult to synthesize true 3-D images with this method.

One significant method for generating the complex amplitude distributions for 3-D objects is ray tracing.[8,9] In this method objects are considered to be composed of a large number of discrete point sources of light, and the complex amplitude of spherical waves from each point source is superimposed on the hologram plane. The ray-tracing method is simple in principle and is potentially the most flexible in synthesizing holograms for true 3-D objects, which should be processed by standard rendering techniques such as hidden-surface elimination or surface shading. However, there is a large computational problem: Calculating an interference fringe pattern requires an enormous computation time. Recent computer technology makes it possible to synthesize holograms on a desktop computer, but even on a state-of-the-art computer it takes several hours or as long as a few days to create a full-parallax hologram by ray tracing. Only a horizontal-parallax-only hologram allows for the quasi-real-time synthesis of 3-D images based on ray tracing.[10,11]

A typical method for reducing computation time in ray tracing is to calculate the real-valued fringe intensity instead of the complex-valued amplitude of the light wave from objects.[9,12] Another method uses geometric symmetry to avoid redundant calculation of the distance between a point source of the object and a sampling point of the hologram.[13] In addition, the use of a lookup table remarkably improves the computation time in ray tracing,[12] but the main disadvantage of a lookup table is the huge size of the table, which increases with the number of sampling points in the object space. In another approach, computer-graphics hardware is used to assist in calculating the diffraction pattern of point sources.[14] In this vein, an attempt was made to fabricate a special-purpose computer for the fast synthesis of 3-D holograms.[15,16] These approaches may be attractive for resolving problems of the real-time synthesis of digital holograms in the future.

The authors are with the Department of Electrical Engineering, Kansai University, 3-3-35 Yamate-cho, Suita, Osaka 564-8680, Japan. K. Matsushima's e-mail address is matsu@kansai-u.ac.jp.

Received 6 December 1999; revised manuscript received 5 September 2000.

0003-6935/00/356587-08$15.00/0

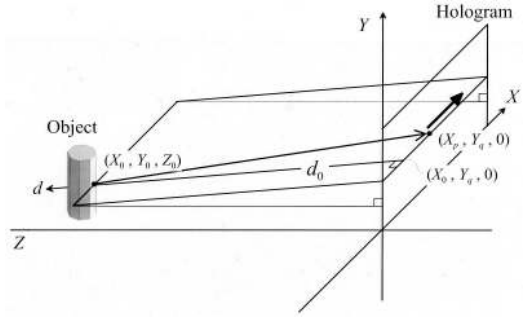

Fig. 1. Diagram of reference coordinate system.

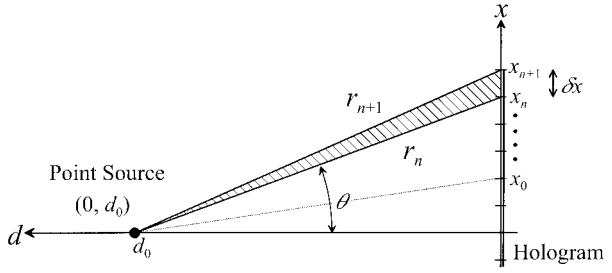

Fig. 2. 2-D coordinates.

In ray tracing, the phase of the light wave from a point source is obtained from the distance between the point source and the sampling point on the hologram. Most of the computation time is spent calculating this 3-D distance, because the calculation requires an arithmetic operation on the square root. Thus an approximation based on a polynomial expansion of the square-root function is sometimes used to generate a fringe pattern rapidly. However, the precision of the numerical calculation in this approximation significantly decreases with an increase in the angle of incidence of the object light with respect to the hologram.

In this paper we propose three types of recurrence formula for fast and accurate computation of the 3-D distance. This algorithm takes advantage of the fact that the difference between the distances from a point source to a sampling point and to the next sampling point is much smaller than the 3-D distance itself.

2. Ray Tracing and Its Computation Sequence

In ray tracing, we treat 3-D virtual objects as if they were composed of a large number of point sources of light. As is shown in Fig. 1, suppose that the position of the jth point source of light is given as (X_j^o, Y_j^o, Z_j^o). A spherical wave emitted from a point source is incident upon the hologram plane. The complex amplitude of the spherical wave is sampled on the hologram plane at intervals of δx in the X direction and at intervals of δy in the Y direction. The sampling position on the hologram plane is represented by $X_p = p\delta x$ and $Y_q = q\delta y$ in the x and the y directions, respectively. Accordingly, the total complex amplitude sampled on the hologram plane is given as[13,14]

$$O(X_p, Y_q) = \sum_{j=0}^{N-1} \frac{a_j}{r_j(X_p, Y_q)} \exp[ikr_j(X_p, Y_q)], \quad (1)$$

where N and a_j denote the total number of point sources and the amplitude of the jth point source, respectively. The distance between the jth point source and the sampling point located at (X_p, Y_q) on the hologram plane is written as

$$r_j(X_p, Y_q) = [(X_p - X_j^o)^2 + (Y_q - Y_j^o)^2 + Z_j^{o2}]^{1/2}. \quad (2)$$

The object light is simultaneously incident upon all the sampling points. In actual sequences in computation, however, the location at which the amplitude is calculated moves the hologram plane in the X or the Y direction. This is similar to the raster of CRT's. The point at the coordinates (X_p, Y_q) shown in Fig. 1 scans the hologram in the X direction as suffix p increases, and this scan line then moves in the Y direction of the hologram plane. Although the exact procedures are dependent on the implementation and may not be identical, this procedure is commonly used in the calculation of the light wave of objects by ray tracing.

In ray tracing, since computers spend almost all of the time computing the 3-D distance represented by Eq. (2), it is important to reduce this computation. Therefore we take advantage of the fact that the 3-D distance at a sampling point hardly changes at the next sampling point.

To simplify the problem, we introduce 2-D Cartesian coordinates on the plane that contains both a point source located at (X_0, Y_0, Z_0) and a scan line at Y_q. As is shown in Fig. 1, we define the x axis parallel to the X axis and the d axis perpendicular to the x axis on this plain. The origin of the 2-D coordinates is at the point $(X_0, Y_q, 0)$ in the 3-D coordinates. Sampling points on the x axis are defined as $x_n \equiv x_0 + n\delta x$ $(n = 0, 1, 2, \ldots)$, and these points are associated with the X coordinates through $x_n = X_{p+n} - X_0$. Hence the 3-D distance between the point source at (X_0, Y_0, Z_0) and a sampling point at $(X_p, Y_q, 0)$ is rewritten as

$$r(x_n) = (x_n^2 + d_0^2)^{1/2},$$
$$d_0 = [(Y_p - Y_0)^2 + Z_0^2]^{1/2}, \quad (3)$$

where d_0, defined above, is kept constant during a scan in the X direction.

3. Recurrence Formula

We assume that the incidence angle θ formed between the d axis and a straight line from a point source to a sampling point on the hologram is small enough to satisfy $x_n \ll d_0$ (Fig. 2), and we apply an approximation based on a polynomial expansion of the square-root function in a calculation of $r(x_n)$. The approximation, referred to as a binomial approximation in this paper, is obtained by use of the first

two terms of the polynomial expansion in the following:

$$r(x_n) \simeq d_0 + \frac{{x_n}^2}{2d_0}. \tag{4}$$

This approximation is suitable for display devices with low spatial resolution, such as liquid-crystal panels, because the incidence angle in these devices is limited to a low value by the sampling theorem to avoid aliasing. In devices not less than approximately $10/\lambda$ in spatial resolution, the incidence angle can become so large that the binomial approximation is no longer appropriate. In that case the numerical error of approximation (4) would significantly increase with increasing x_n. This problem can be avoided by use of a recurrence formula.

Suppose that the sampling pitch δx is sufficiently smaller than the distance $r(x_n)$. The distance $r(x_{n+1})$ can be expanded into a Taylor's series around x_n:

$$\begin{aligned} r(x_{n+1}) &= r(x_n) + \delta x r'(x_n) + \frac{\delta x^2}{2} r''(x_n) + \ldots \\ &= r(x_n) + \left(x_n + \frac{\delta x}{2}\right)\left[\frac{\delta x}{r(x_n)}\right] - \frac{{x_n}^2}{2r(x_n)} \\ &\quad \times \left[\frac{\delta x}{r(x_n)}\right]^2 + \ldots . \end{aligned} \tag{5}$$

Assuming that $\delta x/r(x_n)$ is much smaller than unity, the first two terms on the right-hand side are a good approximation for $r(x_{n+1})$. Therefore the distance at x_{n+1} is given approximately by the distance at x_n:

$$r(x_{n+1}) \simeq r(x_n) + \frac{(x_n + \delta x/2)\delta x}{r(x_n)}. \tag{6}$$

A simultaneous recurrence formula for calculating $r(x_n)$ is obtained by substitution of $x_n = x_0 + n\delta x$ into approximation (6) as follows:

$$\begin{aligned} r_{n+1} &= r_n + s_n/r_n, \\ s_{n+1} &= s_n + c_1. \end{aligned} \tag{7}$$

Here s_n is an arithmetical series equivalent to $n\delta x^2$, and it is introduced to eliminate the multiplication from Eqs. (7). The starter value and the constant c_1 are given as

$$\begin{aligned} r_0 &\equiv ({d_0}^2 + {x_0}^2)^{1/2}, \\ s_0 &\equiv (x_0 + \delta x/2)\delta x, \\ c_1 &\equiv \delta x^2. \end{aligned} \tag{8}$$

In this recurrence formula only two additions and a division are required for each step. Moreover, the numerical error of Eqs. (7) has significantly less dependence on the incidence angle θ, as is described in the next section.

4. Numerical Errors of the Recurrence Formula

Figures 3 and 4 show numerical errors of distances calculated by the recurrence formula. The numerical error is defined as $E_{\text{cal}} = |r_n - r(x_n)|$, and this error is indicated in units of wavelength λ, where $r(x_n)$ denotes an exact distance.

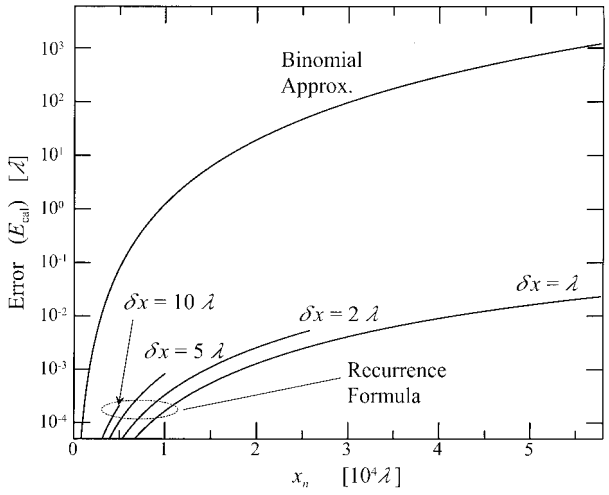

Fig. 3. Numerical error of recurrence formula and binomial approximation; $d_0 = 10^5\lambda$.

In Fig. 3 the numerical errors of the recurrence formula and the binomial approximation of Eq. (4) are plotted on a logarithmic scale as functions of x_n with sampling pitches of the hologram from λ to 10λ. The recurrence formula was evaluated step by step from $x = 0$ to $x_{\max} = d_0 \tan \theta_{\max}$, where $\theta_{\max}$ is the maximum incidence angle, to avoid aliasing. For in-line-type holograms, this angle is given by $\theta_{\max} = \tan^{-1}(\lambda/2\delta x)$.

As is shown in Fig. 3, the error of the recurrence formula is dependent on δx, unlike in the binomial approximation. This error is less than $\lambda/10$ at the $x_{\max}$ with a sampling pitch of λ, whereas at the same location of $x_{\max}$ the polynomial approximation gives rise to an error in excess of $10^3\lambda$.

Figure 3 shows that the precision of the recurrence formula is superior to that of the binomial approximation, and its error is dependent on the sampling pitch as well as on the incidence angle. Therefore in

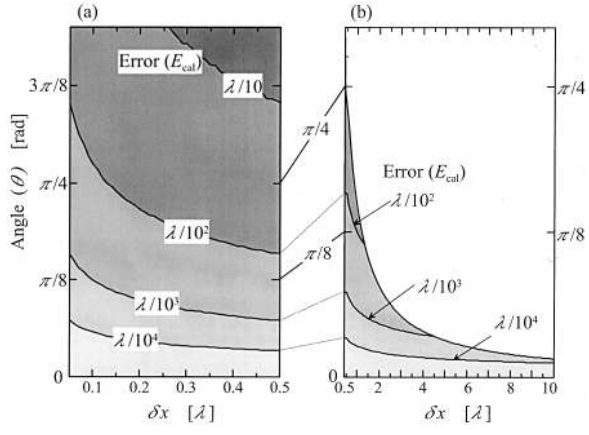

Fig. 4. Error contours given by numerical calculations of recurrence formula for sampling pitches of (a) $\delta x \le \lambda/2$ and (b) $\delta x > \lambda/2$.

Fig. 4 we plotted contours of the numerical error on the parameter plane spanned by the sampling pitches and the incidence angle. Contours for the sampling pitches that are less or more than $\lambda/2$ are shown in the separate Figs. of 4(a) and 4(b), respectively, because the incidence angle is limited to avoid aliasing in the case of Fig. 4(b) but not in Fig. 4(a). As is shown in Fig. 4(a), the error does not exceed $\lambda/10$ for incidence angles less than ~60°. In Fig. 4(b), since the error of the recurrence formula never exceeds $\lambda/10$, there is no contour line for an error of $\lambda/10$.

The numerical error of the recurrence formula given by Eqs. (7) does not exceed $\lambda/10$ in almost all of the area indicated in Fig. 4. However, if a precision of $\lambda/10$ or higher is required for a certain fringe calculation, it is possible to use the algorithm for error control that we present in Section 5.

5. Error Estimation and Error Control

Errors for both the recurrence formula and the binomial approximation can be estimated from remainders of the series expansion.

The polynomial expansion of the square root is written as

$$r(x_n) = d_0 + \frac{x_n^2}{2d_0} - \frac{x_n^4}{8d_0^3} + \dots . \tag{9}$$

The binomial approximation error is given as the third term on the right-hand side of Eq. (9). When $x_0 = 0$, the error expression is rearranged by substitution of $x_n = n\delta x$:

$$E_{\mathrm{BA}} \simeq -\frac{\delta x^4}{8d_0^3} n^4. \tag{10}$$

In the recurrence formula a single-step error $\epsilon(k)$, introduced when x_{k+1} is calculated from x_k, is given as the term $[\delta x/r(x_k)]^2$ in Eq. (5):

$$\epsilon(k) \cong -\frac{1}{2}\frac{\delta x^2 x_k^2}{r(x_k)^3}. \tag{11}$$

Since the starter value of r_0 has no error, the total error after repetition n can be obtained by integration of $\epsilon(k)$ from 0 to n. Therefore the total error is written as

$$E_{\mathrm{RF}}(n) = \int_0^n \epsilon(k)\mathrm{d}k, \tag{12}$$

$$\simeq -\frac{\delta x}{6r_0^3}[(x_0 + n\delta x)^3 - x_0^3], \tag{13}$$

where $r(x_k)$ in the denominator of relation (11) is approximated by the constant distance r_0. In particular, the error of the recurrence formula when $x_0 = 0$ is given as

$$E_{\mathrm{RF}} \simeq -\frac{\delta x^4}{6d_0^3} n^3. \tag{14}$$

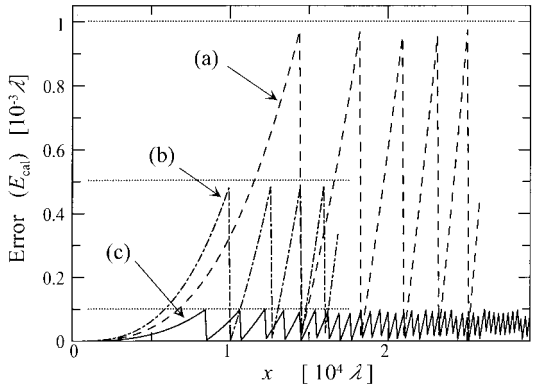

Fig. 5. Error of recurrence formula with error control; $d_0 = 10^5\lambda$. The sampling pitches and permissible errors are (a) $\delta x = 2\lambda$, $E_{\mathrm{lim}} = 10^{-3}\lambda$; (b) $\delta x = 3\lambda$, $E_{\mathrm{lim}} = 5 \times 10^{-3}\lambda$; (c) $\delta x = \lambda$, $E_{\mathrm{lim}} = 10^{-4}\lambda$.

The ratio of the binomial approximation error to the recurrence formula error is given as $E_{\mathrm{BA}}/E_{\mathrm{RF}} = 3n/4$ by relations (10) and (14). When the incidence angle increases, n becomes quite large. Therefore the recurrence formula has a remarkable advantage over a simple binomial approximation.

In the recurrence formula, numerical errors are accumulated for every single step and increase as the repetition times increase. However, it is possible to reset the accumulated error to zero by recalculation of the exact distance before the error exceeds a permissible limit. The error estimation in relation (14) can be used to predict the critical repetition limit.

The critical repetition limit is obtained as

$$n_{\mathrm{lim}} = \left[\left(x_0^3 - \frac{6r_0^3 E_{\mathrm{lim}}}{\delta x}\right)^{1/3} - x_0\right]\delta x^{-1} \tag{15}$$

when relation (13) is solved for n, where E_{lim} denotes the permissible error of the distance.

Figure 5 shows the errors in distance calculations based on the error-control procedure with permissible errors of, Fig. 5(a), $10^{-3}\lambda$; 5(b), $5 \times 10^{-3}\lambda$; and 5(c), $10^{-4}\lambda$. All error curves are controlled with less than the given permissible errors. The overhead for computing n_{lim}, of course, becomes large for small E_{lim} and should not be ignored in terms of the computation time.

6. Variations of the Recurrence Formula

A. Divisionless Recurrence Formula

In most processors division is one of the hardest arithmetic operations. Division is usually several times as long as multiplication. Thus it is significant to eliminate division from the recurrence formula, but it is not worthwhile if such elimination leads to an increase in the error. To eliminate division, we have introduced a variable p_n, defined as the reciprocal of distance r_n. Moreover, p_{n+1} is calculated by a single step of the Newton–Raphson method, which is an excellent algorithm for calculating a reciprocal value.

If ξ is the reciprocal of χ, a single step of Newton–

Raphson method for improving ξ from an estimated value ξ_0 is described with a function defined as $f(\xi) = \chi - \xi^{-1}$:

$$\xi = \xi_0 - \frac{f(\xi_0)}{f'(\xi_0)} = \xi_0(2 - \chi\xi_0). \tag{16}$$

Thus the following recursion relation is given by substitution of p_{n+1}, r_{n+1}, and p_n into ξ, χ, and ξ_0, respectively:

$$p_{n+1} = p_n(2 - r_{n+1}p_n). \tag{17}$$

The recurrence formula of Eqs. (7) is rewritten, with p_n, as

$$\begin{aligned} r_{n+1} &= r_n + s_n p_n, \\ s_{n+1} &= s_n + c_1. \end{aligned} \tag{18}$$

The starter value of p_n is given as $p_0 = 1/r_0$. Other constants are the same as that of the original basic recurrence formula. This modified divisionless procedure involves three multiplications, two additions, and a subtraction.

The error of the divisionless recurrence formula is actually equivalent to that of the basic one, because the Newton–Raphson method has quadratic convergence, and p_n as the estimated value is quite close to p_{n+1} in conventional holograms. When $\delta x = \lambda$ in Fig. 3, for example, the difference between distances calculated by basic and divisionless recurrence formulas does not exceed $2 \times 10^{-7}\lambda$ at the maximum incidence angle. Therefore, Eqs. (18) can be treated in practice as another expression of Eqs. (7).

B. Scaled–Divisionless Recurrence Formula

Here we employ birefined numerical operations as well as variables encoded as double-precision floating-point numbers to investigate the numerical error of the recurrence formula. This is necessary to retain the accuracy of the calculated distance and to avoid rounding errors. However, electronic circuits that handle double-precision floating-point operations are too large and too complex to be used in hardware fabricated especially for the real-time synthesis of holograms.[15,16] Additionally, current microprocessors tend to support some special instruction sets for multimedia data processing; these instructions simultaneously process multiple data encoded in single-precision floating-point operations. These special functions are expected to be used to synthesize holograms in the future.

The number of significant digits must be reduced to let us take advantage of single-precision floating-point operations. Therefore the following scaled variable should be introduced into the recurrence formula:

$$q_n = r_n/r_0 - 1. \tag{19}$$

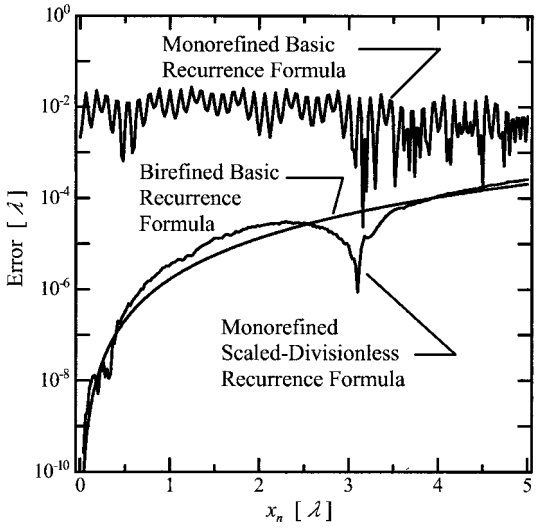

Fig. 6. Error of scaled-divisionless recurrence formula. $d_0 = 10^5\lambda$.

By substituting r_0 from Eq. (19) into relation (18), we simplify the recurrence formula by using q_n to obtain

$$\begin{aligned} q_{n+1} &= q_n + s_n p_n, \\ p_{n+1} &= p_n(2 - q_{n+1}p_n + p_n), \\ s_{n+1} &= s_n + c_2. \end{aligned} \tag{20}$$

The starter formulas and a constant are given as

$$q_0 = 0, \qquad p_0 = 1, \qquad s_0 = (x_0 + \delta x/2)\delta x/(d_0^2 + x_0^2),$$
$$c_2 \equiv \delta x^2/(d_0^2 + x_0^2). \tag{21}$$

Figure 6 shows the error curves of Eqs. (20) with a δx of 10λ. The original basic recurrence formula gives no significant values for monorefined operations, whereas the modified version yields effective distances. However, its error curve involves irregularity owing to the rounding errors.

Unfortunately, because of rounding errors, even the modified version is not useful when δx is less than $\sim 10\lambda$.

7. Measurement of Computation Time

The time required for synthesizing holograms is directly proportional to two factors: the number of point sources for the object and the number of pixels in the hologram. Therefore the computation time of a hologram with M pixels and for an object with N point sources is given as

$$T_{\text{total}} = \tau_a MN, \tag{22}$$

where τ_a is an elemental time coefficient that is dependent on the arithmetic algorithm; τ_a is defined as the time necessary for calculating the fringe intensity of a single point source in a single pixel. We employed the coefficient τ_a in units of nanoseconds per pixel per point to evaluate the performance of the recurrence formulas.

Two processors were used in the performance mea-

surements. One was an Intel Pentium II with a clock frequency of 450 MHz; this is the most common processor and is representative of complex-instruction-set computers (CISC's). Pentium-series processors have an instruction set for calculating the square root and execute it in almost the same time as the division operation. The other processor was an Alpha 21164A with a clock frequency of 600 MHz. This is the typical processor for reduced-instruction-set computers (RISC's) and has no instruction set for the square root. The basic design of the Alpha processor emphasizes the fast execution of simple arithmetic operations instead of the computation of higher functions such as square roots or trigonometric functions.

In the benchmark program we computed the real-valued fringe intensity given as Re[OR^*], where R denotes the distribution of the complex amplitude of the reference light. Since we assume in-line holograms, R is a constant. Thus the bipolar fringe intensity was calculated as

$$I(X_p, Y_q) = \mathrm{Re}[O(X_p, Y_q)] \cong \sum_{j=0}^{N-1} \frac{a_j}{Z_j^o} \cos kr_j(X_p, Y_q), \qquad (23)$$

where we replaced the factor $a_j/r_j(X_p, Y_q)$ in Eq. (1) with a_j/Z_j^o for simplification.

The distance $r_j(X_p, Y_q)$ was computed by use of the square root and three types of recurrence formula, i.e., basic, divisionless, and scaled–divisionless. Bi-refined operations were employed in the square root, the basic, and the divisionless recurrence formulas, and monorefined operations were used in the scaled–divisionless recurrence formula. The benchmark program for both processors was implemented with Microsoft Visual C++ 6.0. This program uses a table lookup method for the trigonometric function but does not employ any reduction method based on geometric symmetry.[13]

We measured the computation time T_N that is necessary for calculating the interference pattern for N point sources in a hologram with 2048 × 2048 pixels. The time coefficient was obtained as $\tau_a = (T_{10} - T_5)/(2048^2 \times 5)$. The time coefficient was determined only for calculating the fringe pattern and did not include overhead time for image scaling, file input–output, or any initialization.

Figure 7 shows the measured time coefficients. The longest time coefficient was 294 (ns/pixel)/point obtained for the square root on the Pentium II, whereas the shortest was 47 (ns/pixel)/point in the divisionless recurrence formula on the Alpha 21164A.

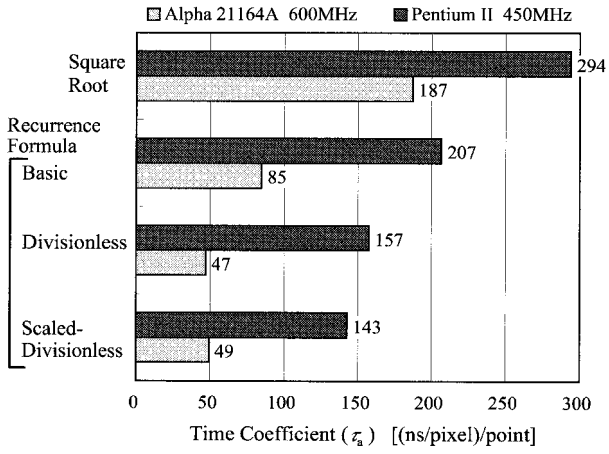

Fig. 7. Performance of recurrence formulas.

8. Construction and Reconstruction of a Hologram

A hologram was synthesized by use of the divisionless recurrence formula. The hologram was an in-line type with binary transmittance. It was fabricated by use of an image setter with a resolution of 4064 dpi. The hologram was composed of 4800 × 4800 pixels with sampling pitches of 6.25 μm in both directions and a dimension of 3 cm × 3 cm. We used the wire frame of a cone as an object, which contains 966 point sources of light. The object has an actual width of 2 cm in the X direction and was set at a Z position of 20 cm behind the hologram.

Figure 8 shows the optically reconstructed image of the fabricated in-line hologram. The total calculation time required for synthesizing the hologram was 1005 s on the Alpha processor. This leads to a time coefficient of 45 (ns/pixel)/point, which is a little shorter than the coefficient shown in Fig. 7. This shorter time coefficient is attributed to our limiting the incidence angle to avoid aliasing, which means that light emitted from a point source would not always cover the whole hologram plane in this setup.

9. Discussion

We used double-precision floating-point variables for most of numerical calculations reported here, be-

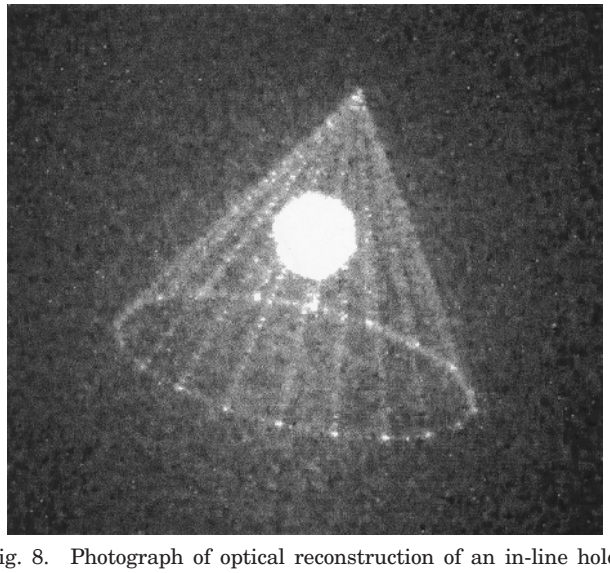

Fig. 8. Photograph of optical reconstruction of an in-line hologram calculated with divisionless recurrence formula. For an object with 966 point sources, calculation takes 16 min 45 s on the Alpha processor.

cause it is expected that the rounding errors may be ignored in double precision. In a rough estimation, the variables for the 3-D distance must be capable of encoding the minimum distance variation of $r_{\min} = r(\delta x) - r(0)$ as well as the distance d_0. Therefore the necessary significant figures are estimated as $\log_{10}(d_0/r_{\min})$. This is rewritten as $\log_{10}(2{d_0}^2/\delta x^2)$ by use of the binomial approximation. For example, the significant figures necessary for a case in which $d_0 = 10^5\lambda$ and $\delta x = \lambda$ are approximately 10.3. In this case, single-precision floating-point computation, which has approximately seven significant figures, does not have enough precision to encode the 3-D distance. For a distance of $d_0 = 10^5\lambda$, the rounding errors in double-precision floating-point operation with 15 significant figures are not significant for $\delta x > 10^{-2}\lambda$.

As is seen from Fig. 7, the performance of the arithmetic algorithm was strongly affected by the architecture of the processors. On the Pentium II processor the shortest τ_a was obtained with the scaled–divisionless recurrence formula, which was approximately half the time of the worst case, the square root. However, on the Alpha processor the shortest time was obtained with the divisionless recurrence formula, which was approximately a quarter of the worst time. The arithmetic operation of the square root in RISC processors is usually implemented as a library provided with the compiler supplier. Although such a library is sophisticated and highly optimized, many steps of the arithmetic operation are needed to guarantee that the calculated value is precision enough for most of the numerical analysis. Therefore the recurrence formula, composed of only a few steps of the arithmetic operation, is more effective in RISC processors than in CISC processors.

We expected the scaled–divisionless recurrence formula with monorefined arithmetic operation to be faster than that of birefined operation only on the Alpha processor, because the Alpha processor has a special instruction set for single-precision floating-point operations. However, the results show that the computation time for the scaled–divisionless formula was approximately 4% longer than that for the divisionless formula on the Alpha processor. This probably corresponds to optimization of the compiler, because the computation time recorded for the scaled–divisionless recurrence formula is sensitive to the setting of the compiler's optimization switches.

10. Conclusion

We have proposed an arithmetic acceleration algorithm that includes recurrence formulas for synthesizing computer-generated 3-D holograms. In the conventional ray-tracing method it is important to reduce the computation time for 3-D distances between a point source of the object and a sampling point of the hologram. The 3-D distance can be calculated with the recurrence formulas in a few simple arithmetic steps.

Three types of recurrence formula have been presented. The basic recurrence formula requires only two additions and a division to obtain the distance. The divisionless recurrence formula, in which the division operation is eliminated by the Newton–Raphson method, is a modification of the basic formula. The precision of both recurrence formulas is sufficient for synthesizing holograms. In addition, the error estimation makes it possible to control the error. In the scaled–divisionless recurrence formula single-precision floating-point operation is allowed for a sampling pitch greater than 10λ. This type of recurrence formula is expected to facilitate the fabrication of special hardware for the real-time synthesis of 3-D holograms.

The performance of recurrence formulas, which was defined as the computation time of the fringe intensity of a single point source for a single pixel, was measured on Pentium II (450 MHz) and Alpha 21164 (600 MHz) processors. The divisionless recurrence formula reduced the time required for synthesizing holograms to approximately a half and a quarter on the Pentium II and the Alpha processors, respectively.

Finally, we constructed an in-line binary hologram with 4800×4800 pixels for an object containing 966 point sources as a demonstration of the performance of the divisionless recurrence formula. The hologram was synthesized on the Alpha processor in 16 min 45 s and was optically reconstructed.

The authors thank Y. Maeda and Y. Yasuda for valuable discussions of mathematics.

References

1. T. S. Huang, "Digital holography," Proc. IEEE **59,** 1335–1347 (1971).
2. G. Tricoles, "Computer generated holograms: an historical review," Appl. Opt. **26,** 4351–4360 (1987).
3. N. Yoshikawa, M. Itoh, and T. Yatagai, "Binary computer-generated holograms for security applications from a synthetic double-exposure method by electron-beam lithography," Opt. Lett. **23,** 1483–1485 (1998).
4. F. Wyrowski and O. Bryndahl, "Iterative Fourier-transform algorithm applied to computer holography," J. Opt. Soc. Am. A **5,** 1058–1065 (1988).
5. F. Wyrowski, "Iterative quantization of digital amplitude holograms," Appl. Opt. **28,** 3864–3870 (1989).
6. M. A. Seldwitz, J. P. Allebach, and D. W. Sweedney, "Synthesis of digital holograms by direct binary search," Appl. Opt. **26,** 2788–2798 (1987).
7. L. Legeard, P. Réfrégier, and P. Ambs, "Multicriteria optimally for iterative encoding of computer-generated holograms," Appl. Opt. **36,** 7444–7449 (1997).
8. J. P. Waters, "Holographic image synthesis utilizing theoretical methods," Appl. Phys. Lett. **9,** 405–407 (1966).
9. A. D. Stein, Z. Wang, and J. J. S. Leigh, "Computer-generated holograms: a simplified ray-tracing approach," Comput. Phys. **6,** 389–392 (1992).
10. P. St.-Hilaire, S. A. Benton, M. Lucente, J. Underkoffler, and H. Yoshikawa, "Electronic display system for computational holography," in *Practical Holography IV*, S. A. Benton, ed., SPIE Proc. **1212,** 174–182 (1990).

11. P. St.-Hilaire, S. A. Benton, M. Lucente, J. Underkoffler, and H. Yoshikawa, "Real-time holographic display: improvements using a multichannel acoustooptic modulator and holographic optical elements," *Practical Holography V*, S. A. Benton, ed., SPIE Proc. **1461,** 254–261 (1991).
12. M. Lucente, "Interactive computation of holograms using a look-up table," J. Electron. Imag. **2,** 28–34 (1993).
13. J. L. Juárez-Pérez, A. Olivares-Pérez, and R. Berriel-Valdos, "Nonredundant calculation for creating digital Fresnel holograms," Appl. Opt. **36,** 7437–7443 (1997).
14. A. Ritter, J. Böttger, O. Deussen, M. König, and T. Strothotte, "Hardware-based rendering of full-parallax synthetic holograms," Appl. Opt. **38,** 1364–1369 (1999).
15. T. Ito, T. Yabe, M. Okazaki, and M. Yanagi, "Special purpose computer holography HORN-1 for reconstruction of virtual image in three dimensions," Comput. Phys. Commun. **82,** 104–110 (1994).
16. T. Ito, H. Eldeib, K. Yoshida, S. Takahashi, T. Yabe, and T. Kunugi, "Special purpose computer for holography HORN-2," Comput. Phys. Commun. **93,** 13–20 (1996).