High-diffraction-efficiency pseudorandom encoding

Yongyi Yang, Henry Stark, and Damla Gurkan

Department of Electrical and Computer Engineering, Illinois Institute of Technology, 3301 South Dearborn Street, Chicago, Illinois 60616

Christy L. Lawson and Robert W. Cohn

The ElectroOptics Research Institute, University of Louisville, Louisville, Kentucky 40292

Received February 9, 1999; revised manuscript received June 21, 1999; accepted July 15, 1999

Pseudorandom encoding (PRE) is a statistics-based procedure in which a pure-phase spatial light modulator (SLM) can yield, on the average, the prescribed diffraction pattern specified by the user. We seek to combine PRE with the optimization of an aperture-based target function. The target function is a fully complex input transmittance, unrealizable by a phase-only SLM, that generates a prescribed light intensity. The optimization is done to increase the diffraction efficiency of the overall process. We compare three optimization methods—Monte Carlo simulation, a genetic algorithm, and a gradient search—for maximizing the diffraction efficiency of a spot-array generator. Calculated solutions are then encoded by PRE, and the resulting diffraction patterns are computer simulated. Details on the complexity of each procedure are furnished, as well as comparisons on the quality, such as uniformity of the output spot array. © 2000 Optical Society of America [S0740-3232(00)01002-4]

OCIS codes: 050.1970, 090.1970, 220.4830, 230.6120.

1. INTRODUCTION

Pseudorandom encoding (PRE) is a procedure that enables a pure-phase spatial light modulator (SLM) to approximately produce the same Fraunhofer diffraction intensity that would result from a desired, but unrealizable, fully complex filter. By a fully complex filter, we mean a device in which *both* amplitude and phase can be varied to produce the desired diffraction pattern.

There is a sizable literature on the design of input generating functions that achieve prescribed far-field intensities subject to a phase-only (PO) constraint.[1–25] Indeed, PRE is only one of many design methods that exist for this purpose. The iterative Fourier transform algorithm is known to give excellent results[4–7,9,10] but requires at least two Fourier transforms per iteration in a procedure that, depending on the complexity of the prescribed diffraction intensity, may extend to hundreds of iterations. Simulated annealing, another powerful method, is likely to yield a global optimum in a phase optimization problem but is reported to be slow[26,27] and thus may be unsuitable for real-time systems that require adaptive redesign of the SLM's modulation pattern.[25] For such systems a noniterative encoding algorithm that requires only a few numerical operations per SLM pixel would seem to be the preferred choice. PRE is such an encoding procedure. However, the overall diffraction efficiency of the process must be reasonably high. If some sort of optimization is required to increase the diffraction efficiency, the additive time component associated with the optimization algorithm becomes a major factor in evaluating its efficiency.

For readers not familiar with PRE, we furnish a brief review in Appendix A. More extensive discussions of PRE appear in Refs. 23–25. Some examples of design by the iterative Fourier transform algorithm are given in Refs. 4–6. Iterative Fourier transform algorithm design from a vector-space point of view is discussed in Refs. 9 and 10. A tutorial discussion of vector-space methods is found in Ref. 28. General approaches to diffractive-optics design are found in Refs. 1 and 2. A discussion of virtual source arrays is furnished in Refs. 20 and 21. Various optimation methods useful in diffractive-optics design can be found in Refs. 20, 22, 26, and 27.

2. DIFFRACTION EFFICIENCY AND SIGNAL-TO-NOISE RATIO

The diffraction efficiency of interest here is the so-called input diffraction efficiency η_{in}, defined by (for simplicity, we use only one-dimensional notation)

$$\eta_{\text{in}} = \begin{cases} \dfrac{1}{L}\displaystyle\int_{-L/2}^{L/2} |g(x)|^2\,\mathrm{d}x & \text{(continuous)} \quad (1) \\ \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} |g(i\Delta)|^2 & \text{(discrete)} \quad (2) \end{cases},$$

where L is the aperture dimension, $g(x)$ is the aperture function that generates the desired far field $G(u)$, i.e., $g(x) \leftrightarrow G(u)$, N is the number of discrete cells, and Δ is the cell size, i.e., $\Delta = L/N$.

A PO array of cells can be modeled by the transmittance

$$g_1(x) = K\sum_{i=1}^{N} \exp[j\phi(i\Delta)]\,\mathrm{rect}\left(\frac{x - i\Delta - \Delta/2}{\Delta}\right), \quad (3)$$

where $\phi(i\Delta)$ is the phase of the ith cell and K is a normalizing constant of no interest at present and is therefore set equal to unity. In PRE the phases $\phi(i\Delta)$ are treated as random variables whose statistics are to be adjusted to produce an approximation to the prescribed diffraction pattern. Replacing $g_1(x)$ with $g_1(n\Delta)$ for convenience, we compute the far-field amplitude, resulting from the uniform illumination of $g_1(x)$, as

$$G_1(u) = \Delta\sum_{n=1}^{N} \exp[j\phi(n\Delta)]\exp(-j2\pi u n\Delta) \quad (4)$$

and its expected value as

$$\overline{G_1(u)} = \Delta\sum_{n=1}^{N} g(n\Delta)\exp(-j2\pi u n\Delta), \quad (5)$$

where we have used the fact that $E\{\exp[j\phi(n\Delta)]\} = g(n\Delta)$ for unbiased estimation of $g(n\Delta)$.

The expected value of field fluctuations, $\sigma_1^2(u)$, is given by

$$\sigma_1^2(u) = \overline{|G_1(u)|^2} - [\overline{G_1(u)}]^2 = N - \sum_{n=1}^{N} |g(n\Delta)|^2 = N(1 - \eta_{\mathrm{in}}). \quad (6)$$

A measure of field variability at spatial frequency u is the signal-to-noise ratio (SNR), given by

$$\mathrm{SNR}_1 = \frac{[\overline{G_1(u)}]^2}{\sigma_1^2} = \frac{[\overline{G_1(u)}]^2}{N[1 - \eta_{\mathrm{in}}]}. \quad (7)$$

Clearly, the larger the SNR is, the less is the variability that one would observe from realization to realization. Equation (7) provides a powerful incentive for raising the diffraction intensity in PRE: An increase in η_{in} from 20% to 60% doubles the SNR. We will observe this phenomenon in the numerical results.

3. LAW OF LARGE NUMBERS

Consider next an array consisting of $2N$ cells, each of width $\Delta/2$. The aperture size and the energy entering the system remain the same as those of an N-cell array with cell width Δ. In this case the far field is

$$G_2(u) = \sum_{n=1}^{2N} \exp[j\phi(n\Delta/2)]\exp(-j2\pi u n\Delta/2) \quad (8)$$

$$= \sum_{n=1}^{N} \{\exp[j\phi((2n - 1)\Delta/2)] \times \exp[-j2\pi u(2n - 1)\Delta/2] + \exp[j\phi(2n\Delta/2)]\exp(-j2\pi u 2n\Delta/2)\}, \quad (9)$$

and its expected value is

$$\overline{G_2(u)} = \sum_{n=1}^{N} [g((2n - 1)\Delta/2)\exp(-j2\pi u n\Delta) \times \exp(j\pi u\Delta) + g(n\Delta)\exp(-j2\pi u n\Delta)]. \quad (10)$$

In regions where $\pi u\Delta \ll 1$ (the useful operating range), $\overline{G_2(u)}$ is approximately given by

$$\overline{G_2(u)} \cong \sum_{n=1}^{N} g((n - 1/2)\Delta)\exp(-j2\pi u n\Delta) + \sum_{n=1}^{N} g(n\Delta)\exp(-j2\pi u n\Delta) \cong 2\overline{G_1(u)}. \quad (11)$$

Likewise, the mean-square value of the noise is

$$\sigma_2^2(u) = \overline{|G_2(u)|^2} - [\overline{G_2(u)}]^2 = 2N - \sum_{n=1}^{2N} |g(n\Delta/2)|^2 = 2\sigma_1^2(u). \quad (12)$$

Hence

$$\mathrm{SNR}_2 = \frac{[\overline{G_2(u)}]^2}{\sigma_2^2} \cong 2 \times \mathrm{SNR}_1. \quad (13)$$

Thus, for the same input energy, a doubling of the SNR is achieved by doubling the number of phase cells. Thus the field variability has been reduced, and it is in this sense that the law of large numbers works for PRE.

4. OPTIMIZATION OF SPOT ARRAYS

We now consider the problem of optimizing η_{in} for the desired fully complex spot-array generator. We first note that a broad class of input transmittances can be written as

$$g(x, \phi) = \frac{f(x, \phi)}{\max_x |f(x, \phi)|}, \quad (14)$$

where $g(x, \phi)$ is the generating or target transmittance, $f(x, \phi)$ is a generating function appropriate for the designated task, and ϕ is a free, real, vector parameter. We also note that, regardless of the value of ϕ, $|g(x, \phi)| \leqslant 1$, as befitting a passive device. Under some circumstances ϕ can be used to optimize the performance of the device. For a spot-array generator that furnishes an array of far-field spots at, say, $\{u_i,\ i = 1,\ldots, M\}$, an appropriate generating function is

$$f(x, \phi) = \sum_{k=1}^{M} \exp[j(2\pi u_k x + \phi_k)], \quad (15)$$

where $\phi = (\phi_1,\ldots, \phi_k,\ldots, \phi_M)^{\mathrm{T}}$. For $\phi = \mathbf{0}$, i.e., each component of ϕ is zero, the η_{in} computation yields approximately $\eta_{\mathrm{in}} = 1/M$. A significant improvement can be obtained by a judicious choice of ϕ. Indeed, inserting

Eq. (15) in Eq. (14) and using Eq. (1) yield an input diffraction efficiency described approximately by

$$\eta_{\text{in}}(\phi) = \alpha^2(\phi)M, \tag{16}$$

where

$$\alpha(\phi) \equiv \frac{1}{\max_x |f(x, \phi)|}. \tag{17}$$

Thus, to maximize η_{in}, we seek to find ϕ^* such that

$$\phi^* = \arg[\max_{\phi} \alpha^2(\phi)]. \tag{18}$$

Equation (18), in words, says that ϕ^* is the vector that will make $\alpha^2(\phi)$ as large as possible. Clearly, from Eq. (16), this will maximize η_{in}. However, an analytic solution to Eq. (18) is not readily apparent. Thus, in the rest of this paper, we consider approximate solutions to Eq. (18). In particular, we focus on the three methods described below: Monte Carlo simulation, genetic algorithms, and gradient descent.

A. Monte Carlo Simulation

In the Monte Carlo simulation a random-number (RN) generator is used to generate a new random phase vector at each trial. This phase vector is then used to evaluate η_{in}. The largest η_{in} and the corresponding phase vector are retained. Two experiments were performed: one of 1000 trials and one of 10,000 trials. The phases and the corresponding diffraction efficiencies are recorded for both experiments. The results are given in Section 6.

B. Genetic Algorithm

Details of the genetic algorithm (GA) used for optimizing a 10-spot array are given in Appendix B. Here we furnish only a summary of the parameters used: initial population size is 25, each member being a 10-component vector of phases, each represented by a 10-bit binary string; fitness scaling parameter is 2; fitness function is η_{in}; initial mutation probability is 0.01; and final mutation probability is 0.005.

C. Gradient Method

A gradient algorithm requires an objective function that we seek to extremize. Such an objective function can be constructed by the following reasoning. Recall Eq. (2) for η_{in}; we wish to make η_{in} as large as possible. Now $\eta_{\text{in}} = 1$ if $|g| = \alpha|f| = 1$ for all x or, equivalently, $|f|^2 = 1/\alpha^2 \equiv \beta$. Therefore, more generally, $\beta - |f|^2$ is the error from the optimum at location x, and

$$e(\beta, \phi) \equiv \sum_{k=1}^{N} [\beta - |f(k\Delta, \phi)|^2]^2 \tag{19}$$

is proportional to the total mean square error over all x. Thus $e(\beta, \phi) > 0$ is a suitable objective function that has to be minimized over β and ϕ. To find β and ϕ that would minimize the objective function, we use the iterative gradient formula:

$$(\beta, \phi)_{k+1} = (\beta, \phi)_k - \gamma \nabla e(\beta, \phi), \tag{20}$$

where

$$\nabla e(\beta, \phi) = \left(\frac{\partial e}{\partial \beta} \frac{\partial e}{\partial \phi_1} \cdots \frac{\partial e}{\partial \phi_M} \right)^{\mathrm{T}},$$

k is the iteration number, γ is a constant step size, and the gradient $\nabla \epsilon(\beta, \phi)$ is evaluated at $(\beta, \phi)_k$. The initial starting vector $(\beta, \phi)_0$ is chosen by using a RN generator to supply values for the components.

5. COMPUTATIONAL EFFICIENCY: COMPARISON

It is of interest to consider how the different optimization routines compare vis-à-vis the amount of computation. In this section we furnish an analysis of the relative computational efforts involved in each of the three optimization algorithms for a 10-spot array.

A. Monte Carlo Search

Let T_ϕ denote the time that it takes to draw 10 random phases and $T_{\max}$ the time needed to find the maximum value of the generating function $f(x, \phi)$. For each phase vector ϕ, we need to compute the diffraction efficiency in Eq. (2). Let $T_{\eta_{\text{in}}}$ denote the time to compute η_{in}. Then the approximate computation time per cycle is

$$T_\phi + T_{\max} + T_{\eta_{\text{in}}},$$

and for C cycles the total time would be

$$T_T^{(\text{MC})} = C(T_\phi + T_{\max} + T_{\eta_{\text{in}}}). \tag{21}$$

In our simulation we tried two values of C: 1000 and 10,000.

B. Genetic Algorithm

For each of the 25 strings, we must draw a 10-component random phase vector and evaluate the associated η_{in}. There are 25 probability computations and 25 scaling operations. Then with T_s, T_{co}, and T_{mu} denoting the time for scaling, crossover, and mutations, the total time for the GA is

$$\begin{aligned} T_T^{(\text{GA})} &= 25Q(T_\phi + T_{\max} + T_{\eta_{\text{in}}} + T_s + T_{\text{co}} + T_{\text{mu}}) \\ &\cong 25Q(T_\phi + T_{\max} + T_{\eta_{\text{in}}}), \end{aligned} \tag{22}$$

where Q is the number of generations required to achieve convergence. We have assumed that $T_s + T_{\text{co}} + T_{\text{mu}} \ll T_\phi + T_{\max} + T_{\eta_{\text{in}}}$. This is borne out by our simulations. The value of Q typically varied near 80.

C. Gradient Search

This case is more difficult to analyze, since the computational load is related to the complexity of the derivative of the generating function. Fortunately, for the 10-spot array, the derivative of the generating function has an analytic form quite similar to that of the generating function itself. Indeed, for the 10-spot array, we can derive that

$$\eta_{\text{in}}(\phi) = \sum_{k=1}^{N} \alpha^2 |f(k\Delta, \phi)|^2$$

$$= \sum_{k=1}^{N} \alpha^2 \sum_{l=1}^{M} \sum_{i=1}^{M} \exp\{j[2\pi(u_l - u_i)k\Delta + \phi_l - \phi_i]\}$$

$$= \sum_{k=1}^{N} \alpha^2 [M + 2Q(k\Delta)], \tag{23}$$

where $Q(k\Delta)$ is given by

$$Q(k\Delta) = \sum_{l=1}^{M} \sum_{i=1+1}^{M} \cos[2\pi(u_l - u_i)k\Delta + \phi_l - \phi_i]. \tag{24}$$

Likewise, we can write the following for the objective function for the 10-spot array:

$$e(\beta, \phi) = \sum_{k=1}^{N} [\beta - M - 2Q(k\Delta)]^2, \tag{25}$$

$$\frac{\partial e}{\partial \beta} = \sum_{k=1}^{N} [2\beta - 2M - 4Q(k\Delta)], \tag{26}$$

$$\frac{\partial e}{\partial \phi_l} = \sum_{k=1}^{N} [2\beta - 2M - 4Q(k\Delta)][2S_l(k\Delta)], \quad l = 1, \ldots, M, \tag{27}$$

where

$$S_l(k\Delta) = \sum_{\substack{i=1 \\ i \neq l}}^{M} \sin[2\pi(u_l - u_i)k\Delta + \phi_l - \phi_i], \quad l = 1, \ldots, M. \tag{28}$$

Comparing Eqs. (27) and (28) with Eq. (23), we see that computation of each gradient component is roughly on the order of $T_{\eta_{\text{in}}}$. The total number of derivative computations is $M + 1$, but since one component of ϕ can be set to zero relative to the others, the actual number is M and the computation time per cycle is $MT_{\eta_{\text{in}}}$. If there are P cycles required for convergence, the total time required is

$$T_T^{(\text{GR})} = PMT_{\eta_{\text{in}}}. \tag{29}$$

In our computations P had the value $P = 20$ to 50.

6. OUTPUT DIFFRACTION EFFICIENCY

While the input diffraction efficiency η_{in} is computed as the energy of the target function, it is useful to have another measure of efficiency that is defined strictly in terms of diffracted light. Such a measure is furnished by the output diffraction efficiency η_{out}, defined by

$$\eta_{\text{out}} = \frac{\text{energy in desired diffraction pattern}}{\text{total energy in frequency plane}} = \frac{\int_R I(u)\,du}{\int_{-\infty}^{\infty} I(u)\,du}, \tag{30}$$

where R is the region containing the desired diffraction pattern. We would like to make η_{out} independent of aperture size, thereby having it reflect the properties of only the generating transmittance $g(x)$. One convenient way to do this is to make the input aperture infinitely large. Thus, in the case of a diffraction pattern consisting of spot arrays, the input aperture would contain an infinitely periodic spatial function whose precise character would depend on the energy distribution among the spots.

One final remark is in order before proceeding. The two efficiencies η_{in} and η_{out} are closely related. This is implied by Eq. (7), which, while it is a result based on an average, shows that as η_{in} goes to unity, essentially all of the diffracted light consists of the prescribed portion $[\overline{G_1(u)}]^2$ as opposed to noise.

7. NUMERICAL RESULTS

Equation (15), with $M = 10$, is used as the generating function for a 10-spot array. The three algorithms—Monte Carlo simulation, genetic, and gradient search—were implemented by using the following parameters: aperture size, $L = 1$; pixel size in the SLM plane, $\Delta x = 0.00195$; pixel size in the Fourier plane, $\Delta u = 1$; number of points N in the discrete Fourier transform, $N = 512$; spot locations in the frequency plane, $u_1 = 210$, $u_2 = 220$, $u_3 = 230$, $u_4 = 240$, $u_5 = 250$, $u_6 = 260$, $u_7 = 270$, $u_8 = 280$, $u_9 = 290$, and $u_{10} = 300$; and number of points in a lookup table of $\sin c^{-1}(x)$, 5000, uniformly spread over the range (0,1). With these data and the material in Table 1, it is possible to replicate all the results in Table 2 and the figures.

Table 1 gives the phase vectors computed for each algorithm. Table 2 gives the performance results for the

Table 1. Optimum Phase Angles[a]

Algorithm	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6	ϕ_7	ϕ_8	ϕ_9	ϕ_{10}
MCS1	0.9729	0.5579	0.7178	0.1777	0.5698	0.3401	0.2559	0.5098	0.5428	0.6863
MCS2	0.3224	0.8596	0.6905	0.2020	0.3622	0.7764	0.3858	0.4821	0.3755	0.3600
GA	0.7243	0.6784	0.6149	0.7331	0.1056	0.4018	0.7185	0.2845	0.9130	0.4770
Gradient	0.6705	0.2675	0.3119	0.8618	0.1963	−0.1426	0.6288	0.8404	−0.0690	0

[a] Final phases obtained by three algorithms: MCS1 is the Monte Carlo simulation with 1000 tries, MCS2 is the Monte Carlo simulation with 10,000 tries, GA is the genetic algorithm, and Gradient is the gradient-search algorithm used with the objective function in Eq. (19). To get the actual phase in radians, multiply each entry by 2π.

Table 2. Performance of Optimization Routines[a]

Performance Measure	Algorithm				
	MCS1	MCS2	GA	Gradient	NPRE
Pseudorandom Encoding					
η_{in} [Eq. (2)]	47.67%	57.98%	64.65%	69.33%	10%
Average η_{out}	47.85%	58.57%	64.96%	69.35%	10.65%
$\overline{\text{ADC}}$ [Eq. (33)]	0.2251	0.1897	0.1597	0.1846	0.5427
ADC* [Eq. (34)] (worst case)	0.3270	0.2924	0.2685	0.2497	0.7942
$\overline{\sigma_N}$ [Eq. (35)]	0.1363	0.1144	0.1010	0.1151	0.3479
Kinoform (Phase-Only)					
η_{out} [Eq. (31)]	94.17%	95.61%	97.96%	96.56%	53.94%
ADC [Eq. (32)]	0.2948	0.3805	0.4616	0.2135	0.9179
σ_N [Eq. (36)]	0.2112	0.2223	0.3013	0.1171	1.3562
Speed	1000	10000	~2000	200–500	0

[a] Input (η_{in}) and output (η_{out}) diffraction efficiencies for the four optimization routines as well as for NPRE and phase-only approaches. The *actual* speed obviously depends on many factors, such as the platform used, the software, and the skill of the programmer. The numbers in the Speed row are the number of cycle times ($T_\phi + T_{\max} + T_{\eta_{\text{in}}}$) required to achieve convergence. For the gradient case the cycle time is actually shorter, since $T_\phi + T_{\max}$ is absent.

three optimization algorithms, as well as for the direct PO (kinoform) result and naive pseudorandom encoding (NPRE). However, before we discuss the results, some remarks are in order:

1. Neither η_{in} nor η_{out} measures the uniformity of the spot intensities in the spot array. Indeed, a large η_{out} implies that most of the light is going to the correct locations but is not indicative that the light is evenly distributed. To control the uniformity of the peaks, one should define a diffraction-plane uniformity metric and use this metric as a constraint in the aperture plane. This is what is done in an iterative vector-space algorithm. We shall call this metric the average deviation contrast (ADC) and define it below.

2. Under ideal circumstances one would expect that, by Parseval's theorem, η_{in} and η_{out} should be essentially equivalent. However, the presence of PRE noise will usually cause a minor difference between them. Since noise intensity adds to the total power in the peaks, it is not surprising to find η_{out} to be slightly larger than η_{in}.

Table 2 needs some explanation. The first row lists the four optimization routines: MCS1 (Monte Carlo simulation with 1000 trials), MCS2 (Monte Carlo simulation with 10,000 trials); GA (genetic algorithm), the gradient search, and, as reference, NPRE, i.e., set $\phi = 0$. The third row lists η_{in} for various cases, with the use of Eq. (2) for η_{in} and Eqs. (14) and (15) for the generating transmittance and the generating function, respectively. The fourth row yields η_{out} for the various cases, with the use of a discrete equivalent of Eq. (30) for the spot array:

$$\eta_{\text{out}} = \frac{\sum_{i=1}^{10} I(u_i)}{\sum_{n=0}^{512} I(n\Delta u)}. \tag{31}$$

Since each PRE trial yields a random outcome, the η_{out} that appears in the fourth row is the average of 10 such trials.

The next three rows are measures of the uniformity of the spots in the spot array. For example, row 5 yields the ADC averaged over 10 trials. The ADC for a single trial, say the ith, is computed as

$$\text{ADC}_i = \frac{I^{(i)}_{\max,p} - I^{(i)}_{\min,p}}{I^{(i)}_{\max,p} + I^{(i)}_{\min,p}}, \tag{32}$$

where $I^{(i)}_{\max,p}$ and $I^{(i)}_{\min,p}$ are the intensities of the highest and lowest peaks at the desired locations, respectively, of the ith trial. A low value of ADC is desired (zero is optimum); a high value indicates considerable variability among peaks. Then row 5 gives

$$\overline{\text{ADC}} = \frac{1}{10}\sum_{i=1}^{10} \text{ADC}_i, \tag{33}$$

while row 6 gives the worst case,

$$\text{ADC*} = \max_i \text{ADC}_i. \tag{34}$$

The entries in row 7 yield

$$\overline{\sigma_N} = \frac{1}{10}\sum_{i=1}^{10} \sigma_N^{(i)}, \tag{35}$$

where $\sigma_N^{(i)}$, the normalized standard deviation, is given by

$$\sigma_N^{(i)} = \frac{\left\{\sum_{n=1}^{10} [I^{(i)}_{\text{peak},n} - \bar{I}^{(i)}_{\text{peak}}]^2\right\}^{1/2}}{\bar{I}^{(i)}_{\text{peak}}}, \tag{36}$$

where $I^{(i)}_{\text{peak},n}$ is the intensity of the nth peak (there are 10) of the ith trial (there are also 10) and $\bar{I}^{(i)}_{\text{peak}}$ is the average peak height of the ith trial, computed as

$$\bar{I}^{(i)}_{\text{peak}} = \frac{1}{10}\sum_{n=1}^{10} I^{(i)}_{\text{peak},n}. \tag{37}$$

In Eq. (36) n is a spatial, transverse index, while i is a longitudinal time or ensemble index. Also, in Eqs. (33)–(35), small numbers are more desirable than large ones.

Based on an examination of the first seven rows of Table 2, it appears that all four optimization routines greatly outperform NPRE. In attempting to evaluate the results, however, it is important to remember that not only is PRE a random process but all the optimization routines contain an element of randomness as well. MCS1, MCS2, and the GA are inherently stochastic procedures, and even the gradient search requires a random starting point. To illustrate the effect of the random starting point on the gradient-search results, we obtained $\eta_{\text{in}} = 68.86\%$ after 20 iterations in one run, $\eta_{\text{in}} = 67.74\%$ after 50 iterations in another run, and $\eta_{\text{in}} = 68.33\%$ in a third run. The values of η_{in} in the third row are the *best* results selected from several trials.

It would appear that the gradient search and the GA clearly outperform MCS1 and MCS2. While the gradient search yields a slightly higher η_{in} than the GA (69% versus 65%), the uniformity of the peaks is slightly better in the GA (ADC = 0.16 versus 0.18 for the gradient algorithm; remember that smaller is better).

The three rows under the heading Kinoform (Phase-Only) yield performance data on the PO case. In other words, having computed the phases by one of the various optimization methods, we realize an input transmittance as $\exp\{j \arg[g(x, \phi)]\}$. Thus the input transmittance is modulated by only the *phase* of the generating function. The most striking observation here is the very high η_{out}. Indeed, it is not difficult to show that, for the PO method, high values of η_{out} are expected.[7] The downside of the PO approach is the possibility of significant unevenness of the peaks in the spot array as compared with the peaks produced by the optimized PRE. The advantage of the latter over the former becomes evident upon comparing rows 4 and 6 with rows 8 and 9, respectively.

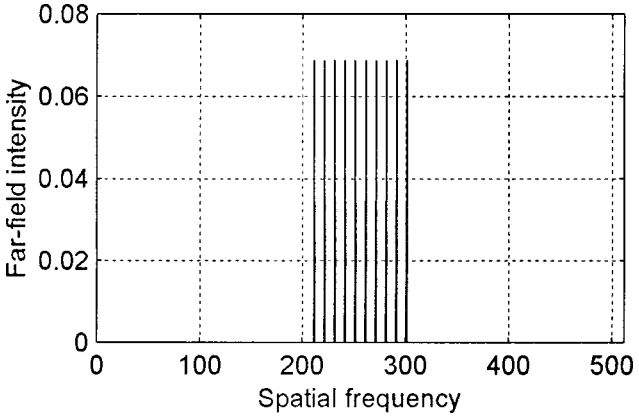

Fig. 1. Diffraction intensity produced by an ideal generating function [Eq. (25)] with no phase optimization.

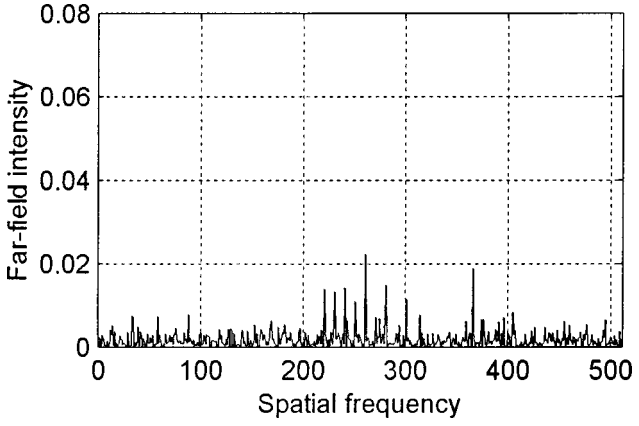

Fig. 2. Diffraction intensity produced by the NPRE algorithm.

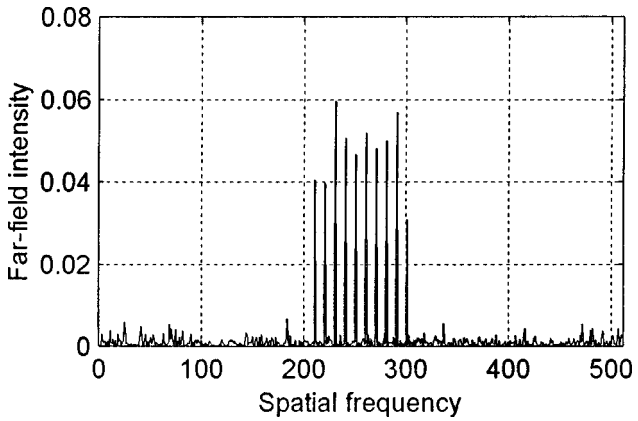

Fig. 3. Diffraction intensity realized by PRE after phase optimization by MCS1.

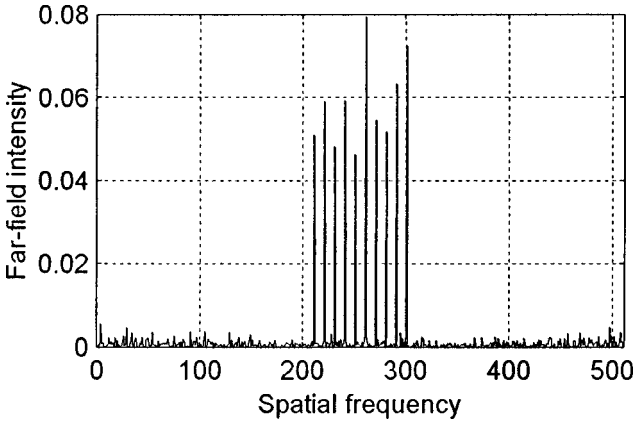

Fig. 4. Diffraction intensity realized by PRE after phase optimization by MCS2.

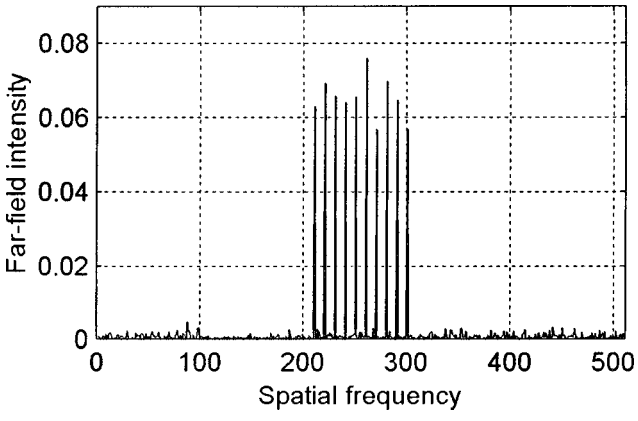

Fig. 5. Diffraction intensity after phase optimization by the genetic algorithm followed by PRE.

Graphical results reinforce the conclusions presented in Table 2. Figure 1 shows the spot array generated by the generating transmittance of Eq. (14) with the use of Eq. (15). In this figure the phases are those computed by the gradient-search algorithm (Table 1). As expected, the peaks are uniform and the cross-term noise (overlapping sidelobes) is virtually imperceptible on a linear scale. Figure 2 shows the "spot array" generated by the NPRE algorithm: The spot array is difficult to detect in the noise; hence the need to optimize the generating function. Figures 3 and 4 show the spot-array peaks with the use of PRE after optimizing by MCS1 and MCS2, respectively. Here the values of η_{out} are 47.71% and 58.85%, respectively. The unevenness of the peaks is greatly reduced over that of the NPRE case but is still pronounced when MCS1 is used. The efficiency in using MCS2 is noticeably higher than that in using MCS1.

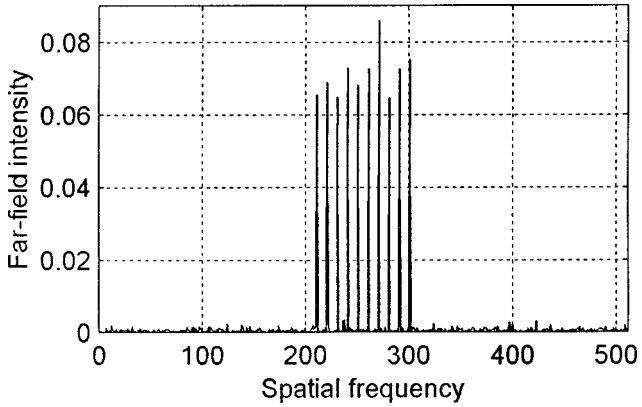

Fig. 6. Diffraction intensity after phase optimization by the gradient algorithm followed by PRE.

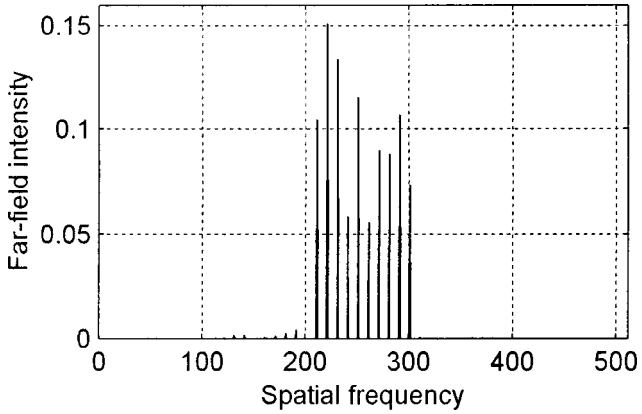

Fig. 7. Phase-only diffraction intensity produced by a genetic-algorithm-optimized generating transmittance.

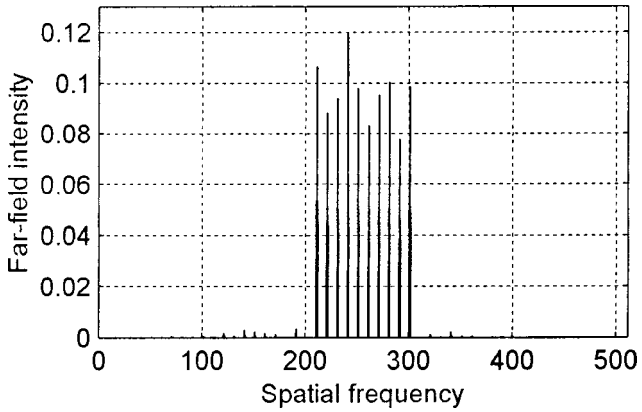

Fig. 8. Phase-only diffraction intensity produced by a gradient-search-optimized generating transmittance.

Figure 5 shows the diffraction intensity furnished by PRE with use of the GA-optimized generating transmittance for $\eta_{\text{out}} = 65.42\%$ and $\bar{\sigma} = 0.0834$. Figure 6 shows the PRE result with use of the gradient-optimized generating transmittance. It has $\eta_{\text{out}} = 71.48\%$ and $\bar{\sigma} = 0.0848$. It can be seen from Fig. 5 that the GA yields the most uniform spot array, while from Fig. 6 we see that the gradient search yields the highest diffraction efficiency.

Finally, Figs. 7 and 8 show the direct PO results with use of the optimized phases from the GA and the gradient search, respectively. As stated above, the diffraction efficiency η_{out} is very high in both cases, but the nonuniformity of the peaks is pronounced in the PO/GA case and might be unacceptable for optical switching or related applications. Interestingly, the PO/gradient-search case gave acceptable results.

8. CONCLUSIONS

Pseudorandom encoding (PRE) is a means for approximately realizing a desired far-field diffraction pattern by modulating only the phase of the input transparency. As a consequence, the prescribed far field is realized, on the average, but in the presence of noise. Optimizing the input diffraction efficiency, the latter being proportional to the energy in the generating transmittance, can reduce the noise and increase the diffraction efficiency. In the case of a spot array, the generating or target function contains an adjustable free-phase vector whose proper selection can lead to higher diffraction efficiencies. Unfortunately, a proper selection by analytic means does not seem possible.

In this paper we considered several techniques for finding the optimum free-phase vector. Best results were obtained by using both a genetic algorithm and a gradient search, which can offer significant improvements over naive PRE and even Monte Carlo simulation. Of the three optimization methods studied, the gradient-search algorithm has the lowest computational complexity.

APPENDIX A: REVIEW OF THE PSEUDORANDOM ENCODING ALGORITHM

There are two key ideas behind PRE: (1) that the expected value of a discrete random variable can be different from any of the values that the random variable can realize and (2) the law of large numbers. How the PRE is affected by the LLN is discussed in Section 3 of the paper. To illustrate the first idea, however, is easy. Suppose that we wish to realize the transmittance value $g(x) = 0.745 \exp(i0.46)$ at a point x but we are limited to a unity-magnitude transmittance and a phase that can take only one of two values: 0 and $\pi/2$. Then with $p = \text{Prob}(\theta = \pi/2)$ and $q = 1 - p = \text{Prob}(\theta = 0)$, we find that $E[\exp(i\theta)] = 0.745 \exp(i0.46)$ when $p = 1/3$. Thus

the transmittance at x has the required value as an *ensemble average* but has a value of either 1 or $j = \sqrt{-1}$ for *any realization*. The mean square error ϵ_{ms} for this example has value $\epsilon_{\mathrm{ms}} = 0.444$. In general, except for some trivial cases, there will always be an error when a fully complex function is realized, on the average, by using an ensemble of PO values.

In the studies of Cohn and co-workers,[23–25] PRE is taken to mean the procedure by which the phases of uniform-magnitude SLM-plane pixels are chosen from an appropriate uniform distribution to achieve an average far-field intensity that approximately corresponds to the prescribed far field.

Realizations of the desired fully complex transmittance function $g(x)$ are achieved through random-sample functions $\{\exp[j\theta(x)]\}$. In particular, at each x, we seek to solve the equation

$$g = \int_{-\infty}^{\infty} \exp(j\alpha) f_{\theta}(\alpha) \mathrm{d}\alpha \tag{A1}$$

for the probability density function $f_{\theta}(\alpha)$, where $g = g(x)$, $\theta = \theta(x)$, etc. A solution to Eq. (A1) is obtained by limiting the solution to the subset of the two-parameter (c, w) family of uniform probability density functions of the form

$$\phi_{\theta}(\alpha; c, w) = \frac{1}{w} \mathrm{rect}\left(\frac{\alpha - c}{w}\right), \tag{A2}$$

where c is the mean and $w^2/12$ is the variance of the associated random variable θ. Using Eq. (A2) in Eq. (A1) yields

$$g = \exp(jc) \frac{\sin(w/2)}{w/2} = \exp(jc)\mathrm{sinc}(w/2\pi). \tag{A3}$$

Then $\exp(j\theta)$ is an unbiased estimator for g if, at the point x, the value of θ is chosen from a uniform probability density function with parameters c and w such that

$$c(x): \quad c = \arg[g(x)], \tag{A4}$$

$$w(x): \quad \mathrm{sinc}(w/2\pi)| = |g(x)|. \tag{A5}$$

In particular, with (c, w) restricted to $0 \leqslant c < 2\pi$ and $0 \leqslant w < 2\pi$ for all x in the support of $g(x)$, the equation $|g(x)| = \mathrm{sinc}(w/2\pi)$ is invertible as $w = 2\pi\,\mathrm{sinc}^{-1}(|g|)$ for $|g(x)| \leqslant 1$.

APPENDIX B: DESCRIPTION OF THE GENETIC ALGORITHM

The steps in implementing the GA for a 10-spot array involve the following:

1. With $\phi_i = (\phi_{i1}, \phi_{i2}, \ldots, \phi_{i10})$ representing the ith trial phase vector, each component ϕ_{ij}, $j = 1, \ldots, 10$, is assigned a 10-bit binary string allowing for the representation of 1024 possible phase values. This is repeated for $i = 1, 2, \ldots, 25$. The choice of 25 population elements is somewhat arbitrary. The actual values of the phases are obtained from a RN generator and converted to binary form. The totality of binary characters representing the phase vector ϕ_i is called a string. The 25 strings, representing the 25 phase vectors ϕ_i, $i = 1, \ldots, 25$, form an initial population.

2. For each ϕ_i, selection probabilities $\{p_i\}$ and cumulative selection probabilities $\{q_i\}$ are computed, respectively, as

$$p_i = \frac{\eta_{\mathrm{in}}^{(i)}}{\sum_{i=1}^{25} \eta_{\mathrm{in}}^{(i)}}, \qquad q_k = \sum_{i=1}^{k} p_i, \tag{B1}$$

where $\eta_{\mathrm{in}}^{(i)}$ is the fitness function of the ith trial phase vector.

3. A new set of fitness values and p_i^*'s and q_k^*'s are created with the use of a linear function to transform the previous set $\{\eta_{\mathrm{in}}^{(i)}\}$ into a new set $\{\eta_{\mathrm{in}}^{(i)*}\}$. The linear function is given by

$$\eta_{\mathrm{in}}^{(i)*} = \alpha \eta_{\mathrm{in}}^{(i)} + \beta,$$

where

$$\alpha = \frac{C\eta_{\max} - \eta_{\mathrm{avg}}}{\eta_{\max} - \eta_{\mathrm{avg}}},$$

$$\eta_{\max} = \max_i \eta_{\mathrm{in}}^{(i)},$$

$$\eta_{\mathrm{avg}} = \frac{1}{25}\sum_{i=1}^{25} \eta_{\mathrm{in}}^{(i)},$$

$$\beta = 1 - \alpha, \tag{B2}$$

and C is a constant determined by the user. Typically, for small populations, C is adjusted to lie in the interval 1.2–2. In our case $C = 2$. The purpose of fitness scaling is to avoid premature convergence as well as maintaining a significant number of high-fitness strings late in the run.

4. With the use of a RN generator to generate numbers in the interval [0,1], a set of 25 strings is selected from the original population according to the newly created cumulative selection probabilities $\{q_k^*\}$.

5. Crossover: Offspring are created from mating pairs randomly selected from the 25 strings. Crossover sites are determined by using outputs of a RN generator issuing integers in the range [1,99].

6. Mutations: These are usually created with very low probabilities. In our case the mutation probability was 0.01 and reduced to 0.005 as the procedure matured.

7. Repeat steps 2–6 to generate subsequent generations. Stop when the convergence criterion has been met.

ACKNOWLEDGMENTS

The work in this paper was supported by Office of Naval Research grant N00014-96-1-1296, a Defense Advanced Research Projects Agency grant through ARO DAAH04-94-G-0358, and NASA cooperative agreement NCC5-222. The authors are grateful to three anonymous reviewers for their perceptive comments, which led to a better paper.

Address correspondence to Yongyi Yang at the location on the title page or by phone, 312-567-3423 or e-mail, yy@ece.iit.edu.

REFERENCES

1. J. N. Mait, "Understanding diffractive optic design in the scalar domain," J. Opt. Soc. Am. A **12**, 2145–2158 (1995).
2. J. N. Mait, "Fourier array generators," in *Micro-Optics: Elements, Systems, and Applications*, H. P. Herzig, ed. (Taylor & Francis, London, 1997), pp. 293–323.
3. U. Krackhardt, J. N. Mait, and N. Streibl, "Upper bound on the diffraction efficiency of phase-only fanout elements," Appl. Opt. **31**, 27–37 (1992).
4. F. Wyrowski, "Diffraction efficiency of analog and quantized digital amplitude holograms: analysis and manipulation," J. Opt. Soc. Am. A **7**, 383–393 (1990).
5. F. Wyrowski and O. Bryngdahl, "Iterative Fourier-transform algorithm applied to computer holography," J. Opt. Soc. Am. A **5**, 1058–1065 (1988).
6. F. Wyrowski, "Iterative quantization of digital amplitude holograms," Appl. Opt. **28**, 3864–3870 (1989).
7. F. Wyrowski, "Upper bound of the diffraction efficiency of diffractive phase elements," Opt. Lett. **16**, 1915–1917 (1991).
8. J. P. Allebach and D. W. Sweeney, "Iterative approaches to computer generated holography," in *Computer-Generated Holography II*, S. H. Lee, ed., Proc. SPIE **884**, 2–9 (1988).
9. H. Stark, W. C. Catino, and J. L. LoCicero, "Design of phase gratings by generalized projections," J. Opt. Soc. Am. A **8**, 566–571 (1991).
10. H. Stark, Y. Yang, and D. Gurkan, "Factors affect convergence in the design of diffractive optics by iterative vector space methods," J. Opt. Soc. Am. A **16**, 149–159 (1999).
11. N. C. Gallagher and B. Liu, "Method for computing kinoforms that reduces image reconstruction error," Appl. Opt. **12**, 2328–2335 (1973).
12. M. P. Dames, R. J. Dowling, P. McKee, and D. Wood, "Efficient optical elements to generate intensity weighted spot arrays: design and fabrication," Appl. Opt. **30**, 2685–2691 (1991).
13. J. Bengtsson, "Kinoform design with an optimal-rotation-angle method," Appl. Opt. **33**, 6879–6884 (1994).
14. B. R. Brown and A. W. Lohmann, "Complex spatial filtering with binary masks," Appl. Opt. **5**, 967–970 (1966).
15. W. H. Lee, "Computer-generated holograms: techniques and applications," in *Progress in Optics*, E. Wolf, ed. (Elsevier, Amsterdam, 1978), Vol. 16, pp. 119–231.
16. W. J. Dallas, "Computer-generated holograms," in *The Computer in Optical Research*, B. R. Frieden, ed. (Springer, Berlin, 1980), Chap. 6, pp. 291–366.
17. L. B. Lesem, P. M. Hirsch, and J. A. Jordon, Jr., "The kinoform: a new wavefront reconstruction device," IBM J. Res. Dev. **13**, 150–155 (1969).
18. D. C. Chu and J. R. Fienup, "Recent approaches to computer-generated holograms," Opt. Eng. **13**, 189–195 (1974).
19. D. Casasent and W. A. Rozzi, "Computer-generated and phase-only synthetic discriminant function filters," Appl. Opt. **25**, 3767–3772 (1986).
20. D. Prongue, H. P. Herzig, R. Dandliker, and M. T. Gale, "Optimized kinoform structures for highly efficient fan-out elements," Appl. Opt. **31**, 5707–5711 (1992).
21. M. W. Farn, "New iterative algorithm for the design of phase-only gratings," in *Holographic Optics: Computer and Optically Generated*, I. Cindrich and S. H. Lee, eds., Proc. SPIE **1555**, 34–42 (1991).
22. J. D. Stack and M. R. Feldman, "Recursive mean-squared-error algorithm for iterative discrete on-axis encoded holograms," Appl. Opt. **31**, 4839–4846 (1992).
23. R. W. Cohn and L. G. Hassebrook, "Representations of fully complex functions on real-time spatial light modulators," in *Optical Information Processing*, F. T. S. Yu and S. Jutamulia, eds. (Cambridge U. Press, Cambridge, UK, 1998), Chap. 15, pp. 396–432.
24. R. W. Cohn and M. Liang, "Approximating fully complex spatial modulation with pseudo-random phase-only modulation," Appl. Opt. **33**, 4406–4415 (1994).
25. R. W. Cohn and M. Liang, "Pseudo-random phase-only encoding of real-time spatial light modulators," Appl. Opt. **35**, 2488–2498 (1996).
26. S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," IEEE Trans. Pattern. Anal. Mach. Intell. **PAMI-6**, 721–741 (1984).
27. S. Kirpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science **220**, 671–680 (1983).
28. H. Stark and Y. Yang, *Vector Space Projections: A Numerical Approach to Signal and Image Processing, Neural Nets, and Optics* (Wiley, New York, 1998).