RL-TR-96-265
Final Technical Report
April 1997

DIGITAL OPTICAL COMPUTING

Alabama A and M University

Anthony Ololo and H. John Caulfield

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Rome, New York

RL-TR-96-265 has been reviewed and is approved for publication.

APPROVED:

RICHARD J. MICHALAK
Project Engineer

FOR THE COMMANDER:

DONALD W. HANSON, Director
Surveillance & Photonics Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE April 1997	3. REPORT TYPE AND DATES COVERED **FINAL, Sep 94 - Sep 95**

4. TITLE AND SUBTITLE
DIGITAL OPTICAL COMPUTING

5. FUNDING NUMBERS
C - F30602-94-C-0251
PE - 62702F
PR - 4600
TA - P3
WU - PX

6. AUTHOR(S)
Anthony Ololo, Dr. H. John Caulfield

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Alabama A&M University
University Eminent Scholar
P.O. Box 1268
Normal AL 35762

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory/OCP
25 Electronic Pky
Rome NY 13441-4515

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
RL-TR-96-265

11. SUPPLEMENTARY NOTES

Rome Laboratory Project Engineer: Richard J. Michalak, OCP, (315) 330-3150

12a. DISTRIBUTION AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*
This effort examined the use of Dammann gratings for optical interconnects within hybrid optoelectronic or all-optical computers. In the effort Genetic algorithm codes were developed in C++. Extensive optimization runs were performed to explore the Genetic algorithm parameter space in the design of efficient Dammann grating devices. Several grating masks were then fabricated using image setting technology.

14. SUBJECT TERMS
optical interconnects, Dammann gratings, optical computing, genetic algorithms, diffractive optics

15. NUMBER OF PAGES
262

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	**UNCLASSIFIED**	**UNCLASSIFIED**	**UNLIMITED**

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

ABSTRACT

THE EVOLUTION OF GENERALISED OPTIMAL OPTICAL INTERCONNECTIONS USING GENETIC ALGORITHMS

This work is aimed at the possibility of using an optimization algorithm to design and then fabricate an optical element that is necessary for the implementation of a digital optical computer and determine analytically for optical interconnection applications, the proper location of a Spatial Light Modulator on an optical processor. For the generation of the optical element, several techniques have been employed in devising this type of binary optical element such as simulated annealing, but in this work, a novel technique and objective function was proposed by us. Here, we proposed to use Genetic Algorithms to design an optical element generally called Dammann Gratings. Dammann Gratings are optical devices that are capable of producing an array of equal intensity light spots from a single incident light wave. For this research effort, I generated 9x9, 17x17, 33x33 and 65x65 array sized Dammann Gratings using various Genetic Algorithm (GA) parameters such as population size, crossover rate, and mutation rate. Also an adaptive mutation mechanism was employed to the various Dammann Grating sizes using a commercial GA package Genesis 5.0 and GenesYs 1.0. The Dammann Grating structures generated in this way were then fabricated as amplitude modulated masks. These masks differ from phase masks only in the "D.C." term (i.e. the signal which occurs in the absence of a mask). The fabrication of the modulated masks was achieved by employing the services of a Graphic Arts Imagesetters. The resulting masks can be optically evaluated using a conventional

"4f" Fourier Transform (FT) lens systems. The results from the optical evaluation are expected to yield 50% of the original incident beam when diffracted through any of the fabricated binary element into the Fraunhofer or the far field region.

KEY WORDS: optical interconnections, genetic algorithms, dammann gratings. spatial light modulators

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Name	Meaning
$\sum$	Sigma	Sum
η	Eta	
δ	Delta	Rate of change
Δ	Delta	Difference
ε	Epsilon	
$\int$	Integral	Summation
Π	Pi	
ρ	Rho	
Ω	Omega	
**		Convolution
∞		Infinity

CHAPTER ONE

INTRODUCTION

1.1 Optical Interconnections

The study and subject of optical interconnections are generating unprecedented interest in the optics community [1]. The interest is a result of the potential advantages to be gained from optics which are absolutely absent from other known methods of information transfer such as electronics. In the area of optical information processing, optics is known to be faster in many cases, to use less power in most, to have a larger space bandwidth product, and to exhibit no electric field interference. With these interesting advantages, come various areas of applications: areas such as digital optical computing, electronic multiprocessors, and optical communications [2]. In the long run, optical interconnection may be expected to replace electronic interconnections in some domains due to speed and the inherent overlapping of beams without interference. This advantage allows for a greater density of interference-free interconnections in free space. One disadvantage of free space interconnection, is the loss of energy which is attributed to diffraction and light scattering. This problem is known to lead to unavoidable cross-talk [3].

Optical interconnections can be divided into four main groups. These are (1) Guided optical interconnections, (2) Free space interconnects, (3) Dynamic interconnects, and (4) Massively parallel dynamic holographic interconnects. In guided optical interconnections, we have applications such as optical fiber bus routing, optical waveguide interconnects, and monolithic GaAs circuits. In free space interconnects, we have applications such as clock distribution to VLSI chips, space-invariant multiple imaging, SLM based interconnection, and Dammann grating illuminators. In dynamic interconnect, we have optical crossbars, optical multistage interconnection, optical clos networks, and optical crossover network. And in massively parallel dynamic holographic interconnects, we have N^4 weighted interconnections and dynamic volume holographic interconnections.

From the various studies already done in this field, clear comparisons of optical and electronic interconnections have been documented [4]. The study [4] established, as shown in figure 1.1, that as densities of interconnections and data rates increase, electronic interconnects approach a limit as set by interline crosstalk and transmission line attenuation. A higher density is achieved with optical interconnect which is limited only by diffraction. In figure 1.2, the study shows the regions of superiority for optical and electronic interconnections. The figure also indicates that optical interconnects have a smaller ratio of power dissipation to data rate than electrical interconnects.

As a contribution to this fascinating and growing topic, I propose to investigate the feasibility of evolving a diffractive optical element (DOE) for optical interconnection purposes by using a nonlinear optimization technique such as Genetic Algorithms as the design tool. My goal in this research effort therefore was: (i) to explore the possibility of using Genetic Algorithms to

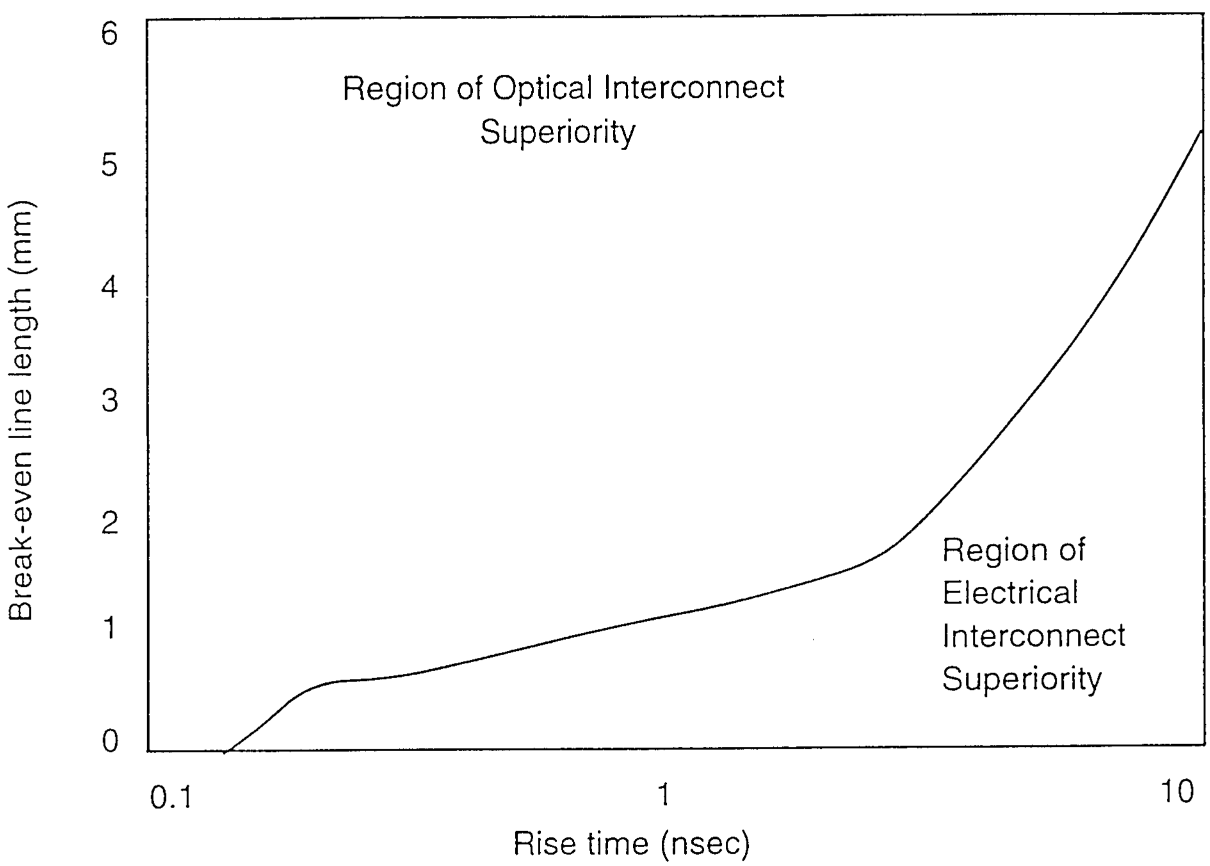


Fig. 1.1 Break-even line length versus Rise time

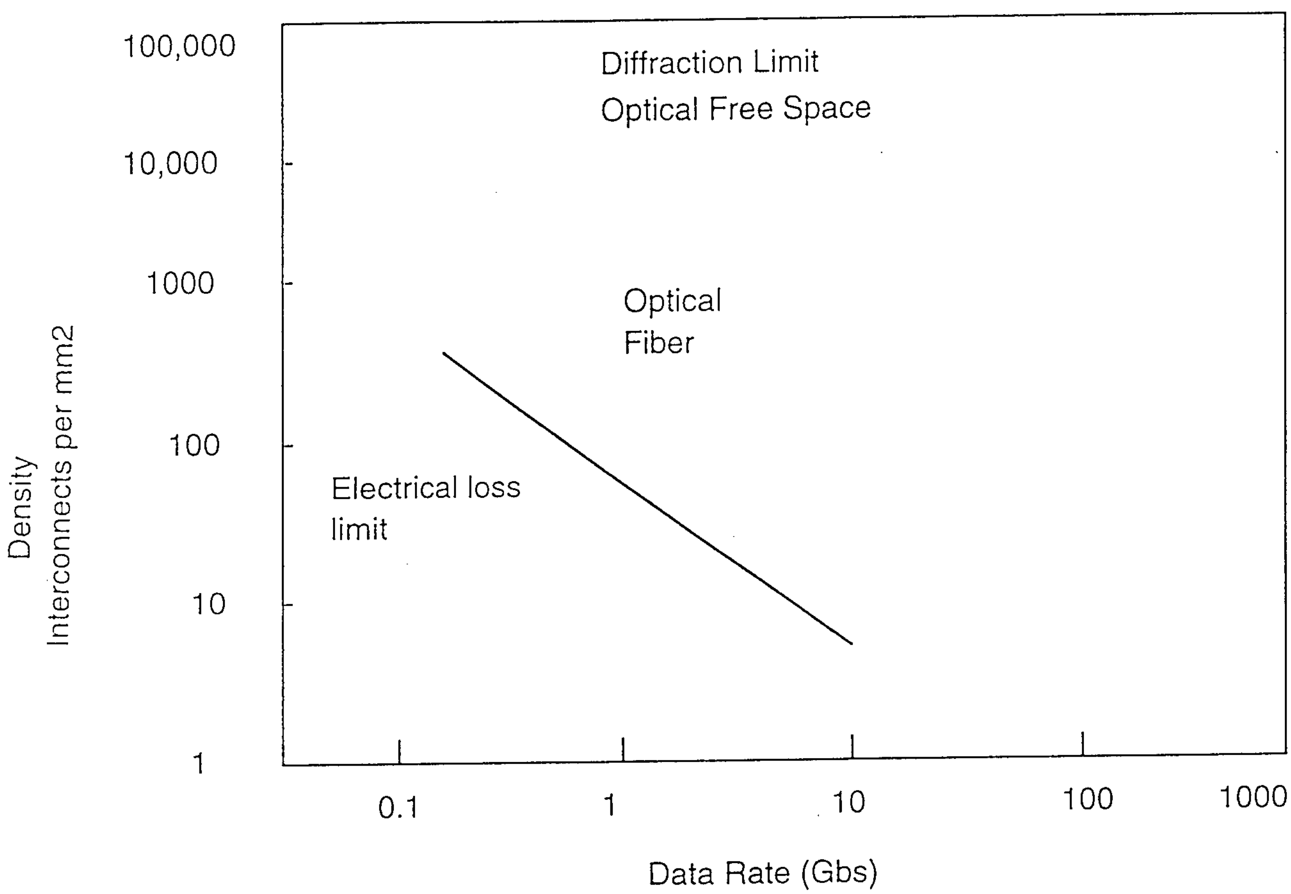


Fig. 1.2 Shows the limits of Electrical and Optical Interconnects.

handle various multiple objective functions, (ii) to evolve various optical architectures using Spatial Light Modulators (SLMs) and analytically determine locations on the optical system the SLM can be placed to yield the desired result. and (iii) to present a computer generated result of the use of Genetic Algorithms as a design tool to make an off-line diffractive optical element such as binary phase Dammann Gratings and any diffractive optical element that is capable of generating an array of equal intensity light spots projected on the far field or the Fraunhofer diffraction region and the fabricated optical element.

1.2 **Dammann Gratings**

Dammann gratings are primarily binary or multilevel phase gratings used to produce a one or two dimensional array of equal intensity light spots from a single incident laser beam [5]. As binary phase gratings, Dammann gratings allow transmitted light through it to be phase shifted by 0 and π. For a phase shift smaller or larger than π, the zeroth diffraction order or dc term may be much brighter than the other elements, a result which is not generally desirable. The array of equal intensity light spots generated from Dammann gratings can then be used, for example, as the optical power supply to bias an array of optical logic devices as illustrated in figure 1.3. Other uses include the making of star couplers, digital optical computing, coherent summation of beams from different laser sources, and space-invariant interconnect. The space-invariant interconnection application is shown in figure 1.4.

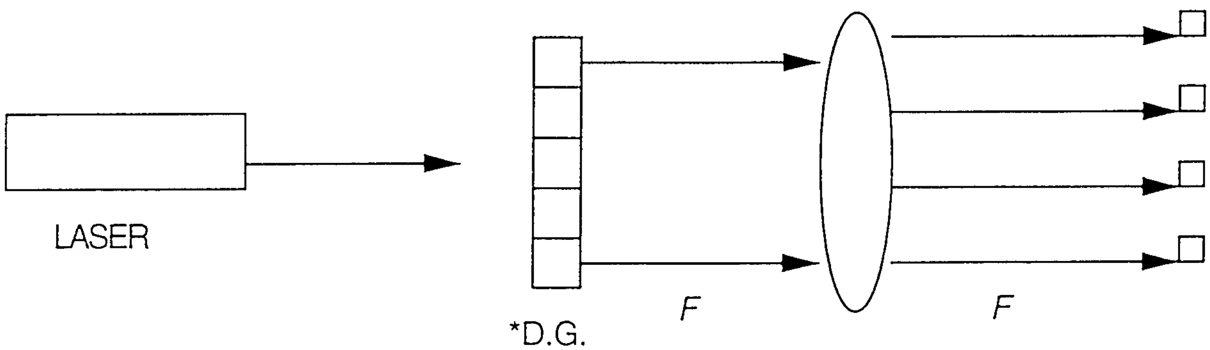


Fig. 1.3 *Dammann Grating as an array illuminator

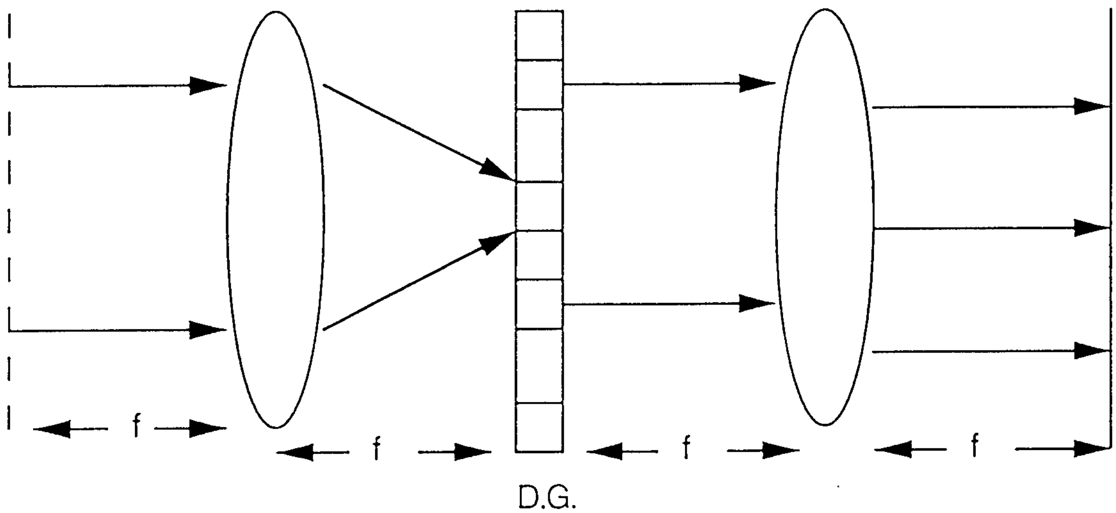


Fig. 1.4 Array generator used as space-invariant interconnect.

The idea of using a special type of diffraction grating to generate an array of equal intensity light spots, came from Hans Dammann, a German holographer, in the early 1970s [6]. His idea later became the first important DOE and any DOEs that could generate uniform array of points of light, and hence came to be known as Dammann Gratings whether or not it applies the formula of Hans Dammann.

For a given single period of a one dimensional Dammann grating which is symmetric and given also that the period p is set to unity $p = 1$, the transmission function will have values 1 and -1 which corresponds to phase values of 0 and π. The consequence of this, is that the grating is completely characterized by its N transition points given as $x_l, \ldots\ldots, x_n$ where the phase changes occur. The transition points or coordinates can be determined by the design algorithm of such design tools as Genetic Algorithms, stimulated annealing, and other known optimization algorithms to yield an array of diffraction orders that are equal in intensity rather than by Dammann's formula. Due also to the symmetry requirements of this type of gratings, the positive and the negative diffraction orders has the same values. Hence, for the first N orders to be equal in intensity to the zeroth order, there will be N equations for the N transition points. For a one dimensional Dammann Grating with N transition points, there are $2N + 1$ diffraction orders, and for a two dimensional grating there are $(2N + 1)^2$ diffraction orders [5]. These may not be the only diffraction orders present. There are also higher diffraction orders occurring in the vicinity due to scattered light. These higher orders compromise the efficiency of the diffracted beams.

A Dammann grating, as a periodic binary-phase function, can be mathematically expressed as [7]

$$F(u) = f(u) * comb(u), \tag{1.1}$$

for a one dimensional grating. Here f(u) is the single period grating function and comb(u) is the sharp peak diffraction pattern. The inverse Fourier transform of the function, which is the response of the grating is given by

$$f(\chi) = \sum_{n=-\infty}^{\infty} f(n)\delta(\chi - n) \tag{1.2}$$

with

$$f(n) = \int_{-1/2}^{1/2} F(u)\exp(j2\pi nu)du \; . \tag{1.3}$$

For a two dimensional Dammann grating which can be obtained by crossing two one dimensional Dammann grating patterns, the expression is

$$F(u,v) = f(u,v) ** comb(u,v). \tag{1.4}$$

The response of the grating is then calculated by

$$f(x, y) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f(n,m)\,\delta(x - n, y - m) \; , \tag{1.5}$$

with

$$f(n,m) = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} F(u,v)\exp[j2\pi(nu + mv]dudv \; . \tag{1.6}$$

For a single period Dammann grating whose positive side is $0<x<1/2$, the amplitude transmission function H(x) is expressed as

$$H(x) = \sum_{n=0}^{N} (-1)^n \mathrm{rect}(x - (x_{n+1} + x_n)/2/(x_{n+1}-x_n)) \qquad (1.7)$$

where rect(x) 1 for |x| 1/2, rect(x) 0, for |x| 1/2 and $x_o = 0$ as defined [8]. By using Fourier transformation, the amplitudes of the diffraction orders can be calculated to be

$$h_o = 2 \sum_{n=0}^{N} (-1)^n (x_{n+1} - x_n), \qquad (1.8)$$

$$h_m = 1/\pi m \sum_{n=0}^{N} (-1)^n [\sin(2\pi m x_{n+1}) - \sin(2\pi m x_n)] . \qquad (1.9)$$

Due to the symmetry property of Dammann gratings, the coefficients are symmetric ($h_m = h_{-m}$) and are real valued ($h_m = h^*_m$). And according to Parseval's theorem, the sum of the intensities of light of the diffraction orders is equal to one:

$$\sum_{m=-\infty}^{\infty} I_m = 1 \quad . \qquad (1.10)$$

The light efficiency can be calculated as

$$\eta = |h_o|^2 + 2 \sum_{m=1}^{N} |h_m|^2 \quad . \qquad (1.11)$$

The minimum feature size, that is, the minimal distance between adjacent transition points is calculated as

$$\delta = \min_{0 \leq n \leq N} |x_{n+1} - x_n| \quad . \tag{1.12}$$

As a measure of the quality of Dammann gratings, it is imperative to calculate a reconstruction error R expressed as

$$\Delta R = \max_{m \in [-N,N]} |1-(2N+1)h_m/\eta, \tag{1.13}$$

where N is the transition points, and h_m the amplitude of the diffracted order m in the array of equal intensity light spots. Reconstruction error is the maximum intensity of light deviation from the ideal intensity.

Other kinds of Dammann gratings are the kinoform structures, quarternary binary phase grating, and the generalized binary phase gratings. Figures 1.5 and 1.6 show the schematic diagram of one dimensional single period even and odd Dammann grating. The two dimensional and kinoform structure Dammann gratings are shown in figures 1.7 and 1.8. Figure 1.9 shows a 9x9 diffraction pattern of equal intensity light spots array.

1.3 **Spatial Light Modulators**

Spatial light modulators (SLMs) are the key elements in the implementation and construction in many optical interconnection systems. SLMs, square or rectangularly shaped, can be electronically or optically addressed. In an optical processing unit, SLMs allow laser beams incident on them to be transformed by a lens to a Fourier plane which may contain a

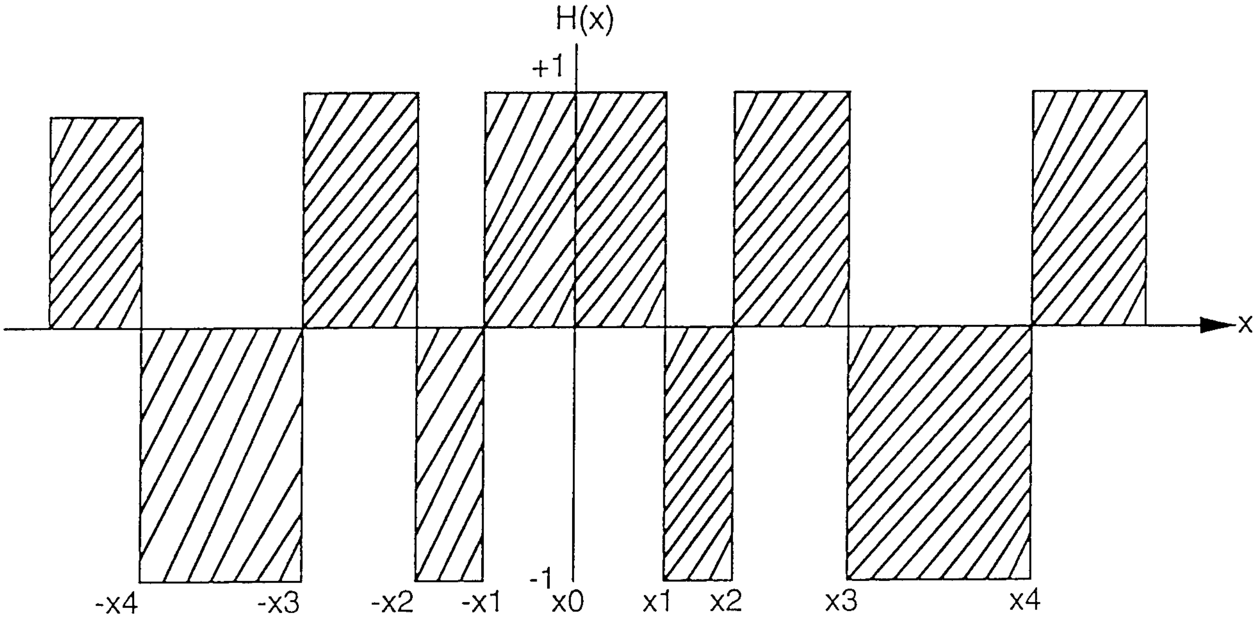


Fig. 1.5 Schematic representation of one dimensional even-ordered Dammann Grating single period.

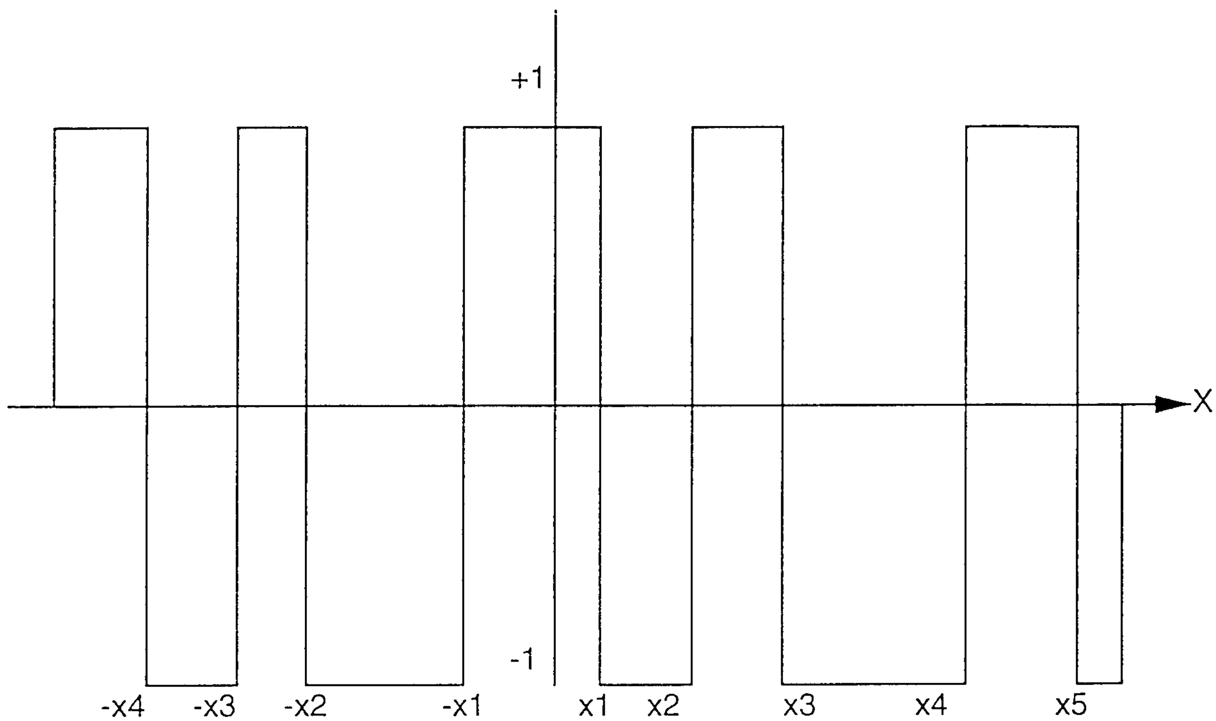


Fig. 1.6 Representation of one dimensional odd Dammann Grating Structure

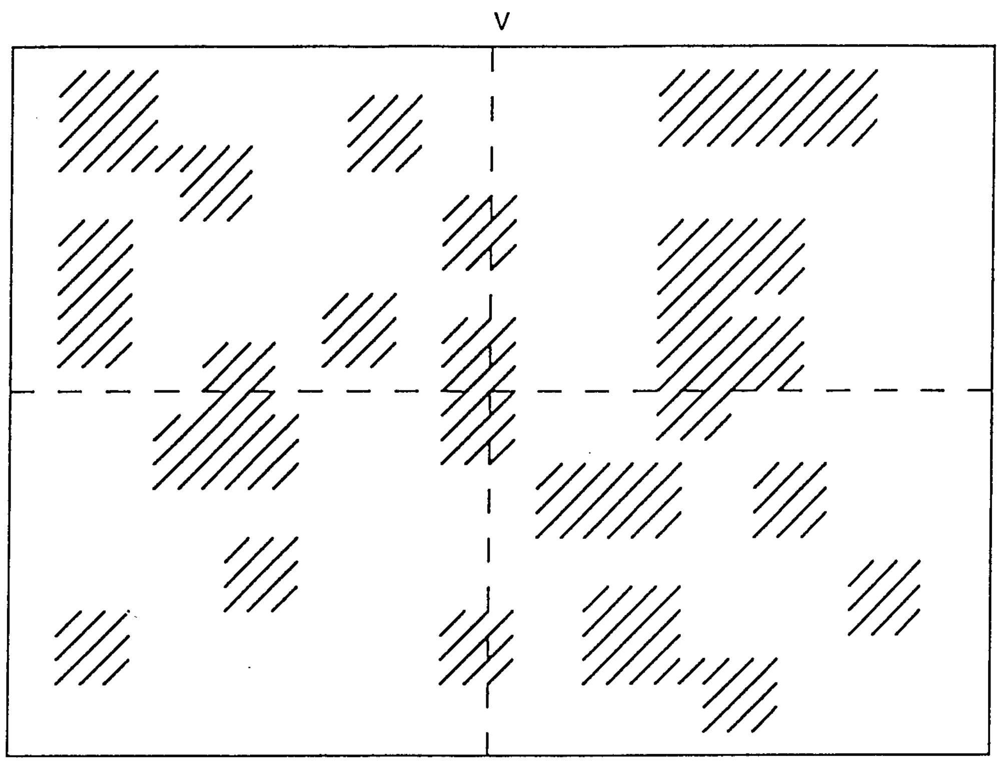


Fig. 1.7 Representation of two dimensional Dammann Grating structure.

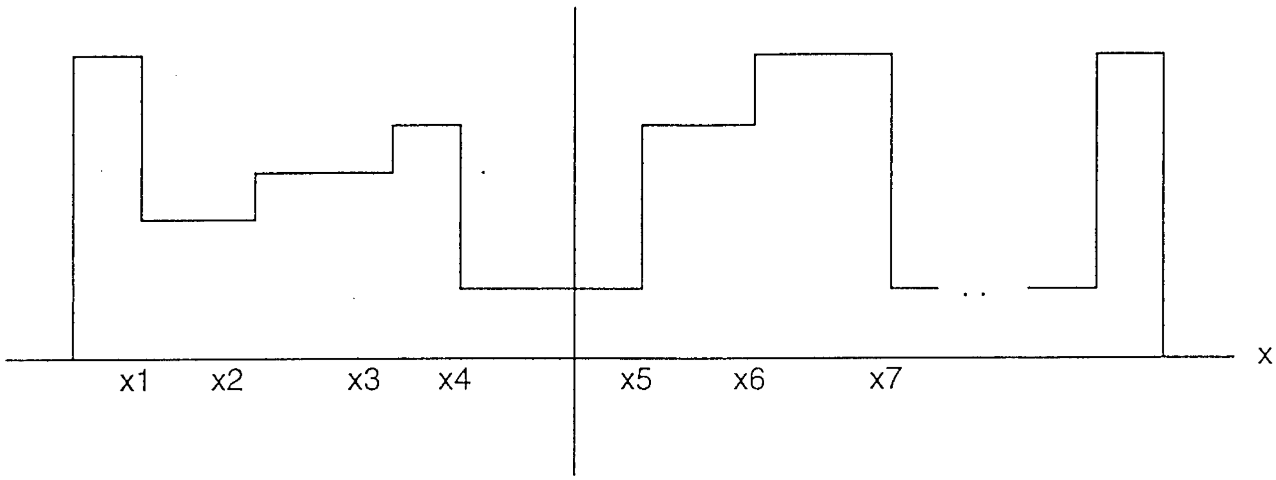


Fig 1.8 Kinoform structure Dammann Grating single period.

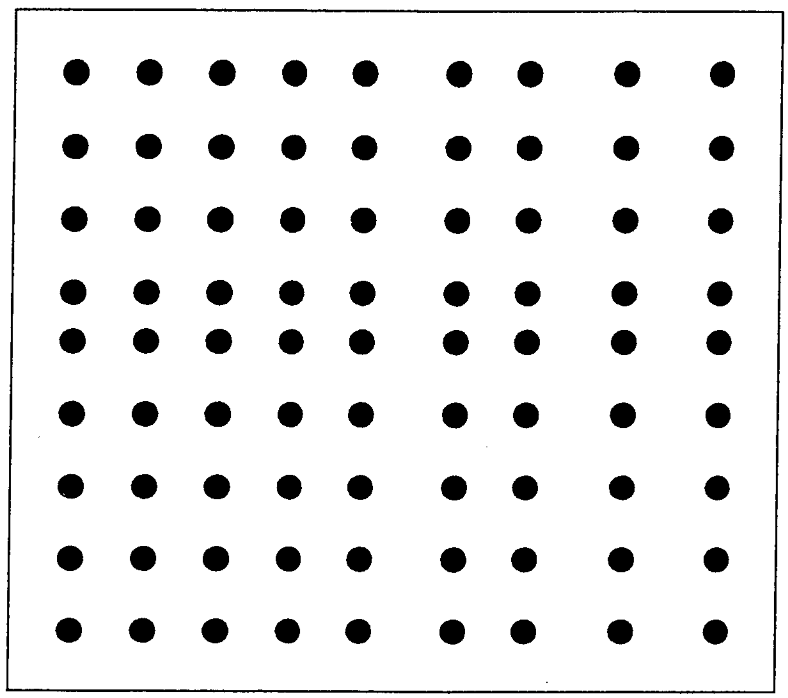


Fig. 1.9 9x9 equal intensity light spot array representation.

myriad of patterns. These patterns are then used as the information to complete an intended optical computation. Typically, SLMs can hold considerable amount of information, up to the order of 10^6 picture elements or pixels. According to Nyquist *et al*, one can anticipate to control up to N/2 x N/2 array of optical beams with an N x N non-negative valued SLM. The uses of SLMs include optical information processing, communications, display systems, miniature liquid crystal TV monitors, large screen projectors for theaters, and flight simulation application [16]. They are also used to perform fundamental operations on parallel information patterns, amplification and signal regeneration, parallel analog arithmetic, and binary logic functions. In artificial neural networks, which are anticipated to revolutionize the computation of complex problems in multiple constraint optimization, pattern recognition, associative memory, adaptive and learning systems, and a host of other challenging artificial intelligence domains such as symbolic processing, vision, speech, and robotics, SLMs are needed to present the optical data that are to be processed, and also to encode stored information. The use of novel optical materials such as quantum wells and ferroelectric liquid crystals allows the construction of very powerful SLMs. SLM technologies today is penetrating into new fields such as phase conjugation, adaptive optics, and neural net systems.

Electronically addressed SLMs essentially can be achieved in at least three ways: (1) electron beams, (2) electrode matrix, and (3) photoconductor. In the electron-beam devices, the modulating material is placed in a vacuum envelope and written on by a scanned electron beam, much like the phosphor screen in a cathode ray tube. Using the electrode matrix.scheme, individual pixels are addressed at the intersections of two perpendicularly- crossing linear arrays of electrodes. The crossed electrodes can either be on opposite faces of the modulating material, or in a single plane. Matrix addressing offers the choice of loading an entire row of information simultaneously to obtain a frame update time equal to the material response time multiplied by the number of rows. In the photoconductor addressing method, photoconductor devices are employed at the junctions of a crossed matrix to provide nonlinearity [17].

Optically addressed spatial light modulators adopt the basic sandwich structure as shown in figure 1.10. In operation, a bias voltage applied to the sandwich is shunted within the illuminated regions of the photoconductor to a voltage-controlled phase, amplitude, and/or polarization modulating material such as electrooptic materials. The essential elements of the optically addressed SLM are the photoconductor or photoreceptor and the modulating material which are separated by a dielectric mirror. The input image activates the photoconductor, which produces a corresponding charge image across the modulator that provides the electric field for the readout material [18].

1.4 **Optical Clock Signal Distribution**

Clock signal distribution is another significant and practical application of free space optical interconnections [8]. Optical clock signal distribution and wafer scale broadcast are quite useful in sending information to various parts of a multiprocessor system all at the same time, thereby ensuring synchronization and fast operation of the system [9]. Variations in the arrival

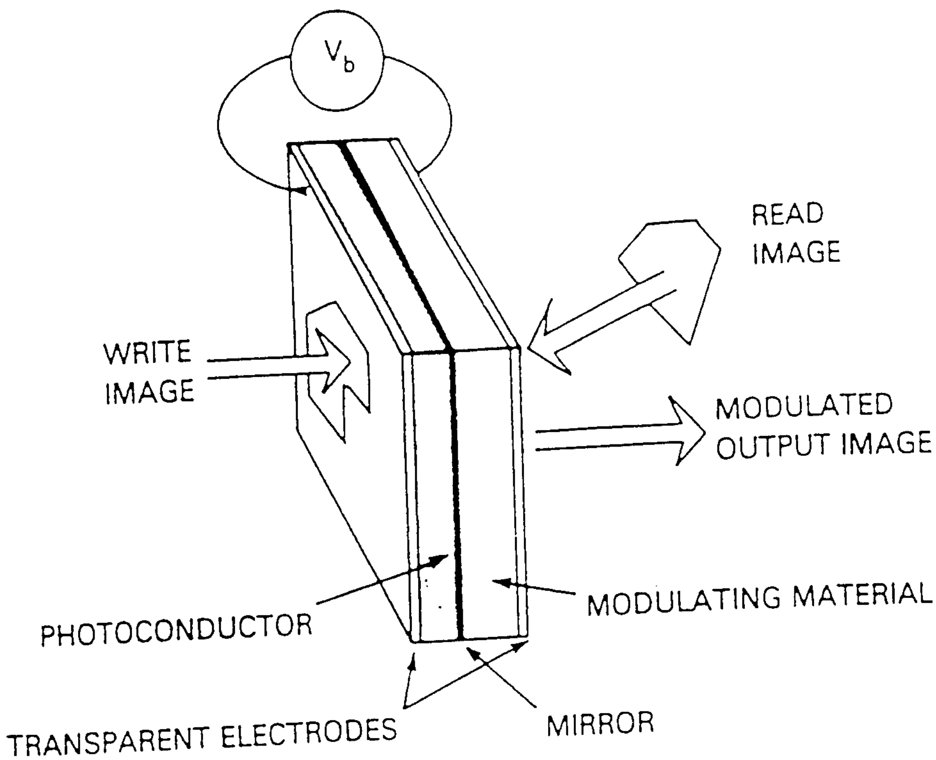


Fig. 10 Electrically Addressed Spatial Light Modulator

times of clock signal can cause problems known as "clock skew". Clock skew is capable of slowing down and causing serious errors in the entire system. This problem can arise due to variations in the run lengths of the clock distribution system, nonuniformity of the signal distribution, and nonuniformity of the detectors [10].

Several methods of using optics to implement clock signal distribution and wafer scale broadcast system are available. Methods such as the use of fiber optics, integrated waveguide optics and free space optics. In this study, we will employ the use of free space solution which may also be extendible to the implementation of parallel high-density data links between various chips and the clocking links.

An optical clock distribution system generally will comprise of an input signal source such as a laser which generates and broadcasts periodic signals of light pulses to various detectors and a diffractive optical element designed to implement the clock distribution network. Figure 1.11 shows a schematic diagram of clock distribution network employing the free space method. Other methods of implementing clock signal distribution are shown in figures 1.12 and 1.13 respectively.

1.5 **Genetic Algorithms (Optimization Technique)**

The design tool or optimization method we will employ in this study to evolve an accurate diffractive optical element such as Dammann Grating and Optical Clock Signal Distribution, is Genetic Algorithms. Genetic Algorithms is a stochastic search algorithm based on the mechanics of natural selection

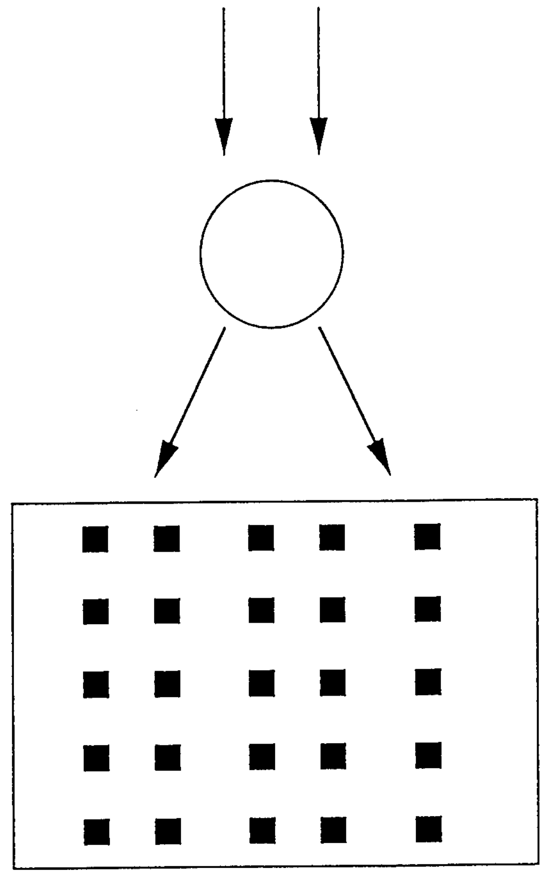


Fig. 1.11 Optical clock distribution technique using free space.

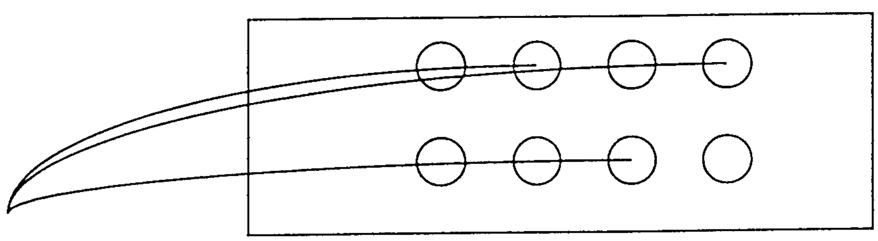


Fig. 1.12 Fiber optics technique for optical clock distribution

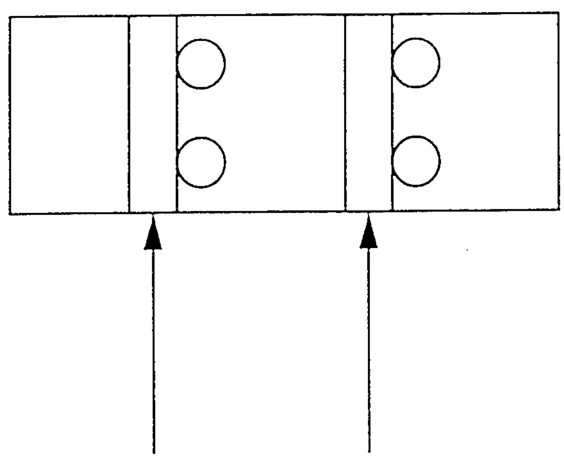


Fig. 1.13 Integrated waveguide optical clock distribution technique.

and evolution. Genetic Algorithms was developed by John Holland and his colleagues in 1975 working at the University of Michigan. In developing this algorithm, Holland and his colleagues had two goals in mind: (1) to explain the adaptive processes of natural systems, and (2) to design artificial systems software that retains the important mechanics of natural systems [11]. The beauty of Genetic Algorithms is the fact that it is robust. That is it strikes a balance between efficiency and efficacy, two necessary factors required for survival in many different environments.

According to John Holland, a Genetic Algorithm can be expressed with the following 8-tuple:

$$GA(P^{\circ}, S, l, s, \rho, \Omega, f, t) \qquad (1.14)$$

where

$P^{\circ} = (X_1, \ldots\ldots X_S)$	initial population
S	population size
l	length of individual's representation
s	selection operator
ρ	operator determination function
Ω	genetic operator set
f	fitness function
t	termination criterion

P°, generally, is the randomly or heuristically generated initial population. The S and *l* parameters describe the number of individuals representing one generation and the length of the genetic representation of each individual, respectively. s the selection operator, produces an intermediate population P'^t from the population P^t by the generation of copies of elements from P^t: $P'^t = s(P^t)$. This is accomplished by taking S subsequent samples from $P^t = (x^t_1 \ldots x^t_S)$ as dictated by the probability distribution p_s. ρ determines an operator w^t_i for each individual $x'^t_i \in P'^t$ which will be applied to this individual. Ω the genetic operator set includes crossover and mutation. The stochastic elements of these operators (application probabilities, e.g. $p_m = 0.001$ and $p_c = 0.6$) [12]. The fitness values are obtained by the fitness function f, and t designates the termination criterion.

As a random search algorithm, Genetic Algorithms can be applied in many facets of our practical day to day activities: in business decisions, science and engineering problems, and in social sciences. Genetic Algorithms solve problems such as finding the best value of x for a function f(x) by first generating an encoded population of candidate solutions. These candidate solutions are described in a form of of structured string of ones and zeros called chromosomes. Several of these strings of ones and zeros make up a population of the candidate solutions. Each 1 or O bit is called genome while the whole string is called a phenome.

With the string of population of binary numbers, one proceeds to apply simple genetic algorithm operators. These are:

Reproduction,

Crossover, and

Mutation.

The process of reproduction accounts for individual strings being copied according to a defined objective function. Strings with higher objective function values have higher probability of being copied, hence are eligible to contribute one or more offsprings in the next generation of new population of candidate solutions. These strings therefore would be selected and allowed to mate (in pairs) at a randomly chosen crossover site. Two new strings are formed from this coupling of each pair of the previously selected strings.

For a pair of strings such as

AI = 1110100011 | 001111

A2 = 0011101011 | 011001

with high fitness values, the crossover site, an integer m is selected at between (*l*, *l* -1). *l* is the length of the string. For the above strings, the crossover site is chosen as the vertical lines shown above. The two new strings formed are:

BI = 1110100011011001

B2 = 0011101011001111

The crossover type used here is the 'one-point' crossover process. Two-point and uniform crossover points are among the other types.

The strings with lowest fitness values are replaced in the next generation by the offsprings. This process is repeated many times until the best result is obtained.

The mutation operator is the occasional random change of the value of the string position. In other words, an intentional change of a 1 bit to a O bit and vice versa. The rate at which mutation occurs is quite small. This is so in order that the strings remain less diversified and reduce noisy signals.

The fundamental theorem guiding the principle and applications of Genetic Algorithms is the Schema theorem [13]. For fitness proportionate reproduction, simple crossover, and mutation, the expected number of duplicates k of a hyperplane of schema H is given by

$$k(H,t+1) \geq k(H,t)f(H)/f\,[1-p_c\,\delta(H)/(l-1) - p_m o(H)]. \quad (1.15)$$

Here f(H) is the hyperplane or schema average fitness, f is the average fitness of the population, p_c and p_m are crossover and mutation probabilities respectively. $\delta(H)$ and o(H) are defining length and order of the schema respectively. l is length of each individual string. In Genetic Algorithms, $\delta(H)$ is the distance between the outermost defining positions of a hyperplane or schema. For example, the defining length of the schema 011*01**, has a value $\delta=5$, i.e. 6-1=5. The asterisk or star *, is a don't care or wild card symbol which corresponds to either a 0 or a 1 at a particular position. The order of the schema o(H), is its fixed number of positions. The schema

011*01** has order equals to 5 and 1*1***** has order of 2. The schema average fitness f(H) is calculated as:

$$f(H) = \sum_{si \, \varepsilon \, H} f(si)/k(H,t), \qquad (1.16)$$

k(H,t) is the expected number of representatives of the hyperplane H. A careful examination of the schemata theorem, leads one to conclude that schemata with fitness values above average will gain an increasing number of trials or sample in the successive generations, while schemata with values below average will undoubtedly receive decreasing number of trials.

1.6 Fast Fourier Transform

The importance of this subject in this work can not be over emphasised. A detailed treatment of the subject will not be carried out in this research as this has already been done by experts in the subject. A Fast Fourier Transform (FFT) will be used extensively in this work, especially in the computation of the various elements involved in this research such as the computation of the power spectrum of the diffractive optical element, Dammann Gratings.

Generally speaking, FFTs are computational tools used by researchers to compute very large data. The FFT algorithm came in existence in the mid-1960s as a consequence of the work of J.W. Cooley and J.W. Tukey. The many algorithms developed for the purpose of computation, is available for FFT computation of Real Functions, Sine and

Cosinė Transforms, Convolution and Deconvolution, Correlation and Autocorrelation, Optimal (Wiener) Filtering, and Power Spectrum Estimation [19].

1.7 Literature Review

Research to derive accurate and efficient diffractive optical elements for optical interconnection systems for applications in digital optical computing is still going on. The use of devices such as binary and multilevel phase gratings for multiple image generation and as array illuminators is of great interest. The earliest computer designed array illuminator or multiple image generation was first derived by Hans Dammann [6]. Dammann used a multiple phase hologram inserted into a conventional optical imaging system to obtain instead of a normal single image, a central block of equally bright images. In this research, he found that using commonly recorded holograms, multiple images were generated instead of array of point light sources. He also found that the holograms were less efficient due to the low reconstruction efficiency of the recorded thin holograms. The multiple images generated, he found were formed as off-axis images which in practice always leads to aberrations. To overcome this drawback, Dammann *et al* [6] suggested the use of high efficiency, inline, phase-only holograms. Using the multiple phase-only hologram, they generated a 15x15 multiple images with 41% of the total radiant flux.

.J. Turunen and colleagues researching in this field, and using a nonlinear optimization methods (simulated annealing and damped least-squares) advanced the technology by calculating array of equal beam size of N=53 with diffraction efficiencies of 80% and 65% for one dimension and two dimension Dammann Grating respectively [14]. The gratings were reported to have low non-uniformity error in the order of few per cent given a grating period of I mm. They also suggested that the Dammann Grating and the focusing lens be combined into a single element. That with, this technology, unwanted high diffraction orders and zero order beam can be filtered off spatially, and hence allow diffraction efficiency of the resulting elements close to 100%. They noted also, that to increase the number of spot arrays beyond 50, it is imperative to multiply the copied holograms resulting from copying the wavefront emerging from the single combination of Dammann Grating and the focusing lens on a thick holographic material such as dichromated gelatin in an array. Their view is that, using this method, there is the possibility of generating N=1000 array of light spots, a size required in parallel optical computers. In another study done by F.B. McCormick of AT&T Bell Laboratories [15] demonstrated a simple technique for generating large (lOOxlO0+) arrays of uniform intensity light spots with good contrast by using Binary Phase Gratings (BPG). The large size array is produced by cascaded BPGs in which the first BPG (or pair of BPGs) forms a small array of spots and the next BPG multiply the images of the small array to form the large array. The reported diffraction efficiency for the combination of four BPGs is of the order of 24.6%. His analysis of the performance of the BPGs, leads to conclusion that inferred that using BPGs with a few transition points, each contributing relatively large amount of error, seems to offer better performance than BPGs with many transition points, with each contributing a small amount of error. For this reason, he said, the multiple imaging scheme offers better spot array uniformity, especially when producing large light spot arrays.

On the design of one dimensional Dammann gratings with phase shift of $\theta = \pi$, H. Lupken *et al* [20] discussed the problems concerning the control of design algorithms which have deterred the progress of advancing the design of Dammann gratings. In this work, they showed how design theory of diffractive elements can be used to avoid these difficulties and how to formulate a straightforward design method to derive a grating optimized in diffraction efficiency and low reconstruction error or uniformity. And by employing the design theory of diffractive elements $\Delta\eta = \eta_1 - \eta$, Lupken and his colleagues designed various Dammann gratings M=15, 33, 53, 101 all with high diffraction efficiency and low reconstruction low. Kwak and his colleagues [21], designed a 9x9 Dammann Grating somewhat optimized by the Newton-Raphson method. Their primary contribution on this on going research effort, was their use of photoinduced anisotropic materials for fabrication which they concluded was easy compared to fabrication process involving common etching processes.

Work on synthetic diffractive optical elements, traditionally known as computer-generated holograms was done by Mohammad *et al* [22]. They inferred that synthetic DOEs might play important role in most optical computing and photonic switching demonstration circuits. In this case, they perform functions such as array illumination, fan-in, and optical interconnection between logic element arrays. At their institution, they designed and fabricated a Fourier-type synthetic DOEs separable Dammann gratings with fan-out up to 128 X 128 with efficiency of 65% and uniformity or reconstruction error of 10%. Also, they fabricated a large number of non-separable trapezoidal designs with 32 X 32 spot arrays.

Since my work generally is dependent on the exploration of the use of Genetic Algorithms for the generation of solutions for binary optical element, we present reviews on previous works done by other researchers employing Genetic Algorithms. Uri Mahlab and colleagues, reported in their 1991 publication, the use of GAs for the implementation of Optical Pattern Recognition [23]. The work centered on the discrimination of two sets of patterns by generating a filter that produces a strong and narrow peak for patterns of the first class and a uniform distribution for patterns of the second class. Goldberg reported in his book "Genetic Algorithms in Search, Optimization, and Machine Learning", the optimization of Pipeline systems using Genetic Algorithms[11]. Other works reported by Goldberg in his book stated above, is the optimization of building structures and medical image registration using GAs. From this work and others, it infers that GAs are quite adaptable to various problems.

In carrying out a feasibility study of Dammann gratings, in which several parameters important for the computation and fabrication are considered, Jahns *et al* computed and fabricated a 40 x 40 array of light spots [24]. From their study, they concluded that array sizes above 40 x 40 present immense problems in computation and resolution. Using simulated annealing and the greedy algorithm techniques, Taghizadeh and colleagues, found a solution to a grating structure with fan-out as large as 201x201 [25]. In the report, they inferred that using the technique large array structures can be calculated, but with increase in computation time using personal computer. In this study also, it was revealed that fabrication materials such as glass and thin films of photoresist, suffer disadvantages such as difficulty in creating accurate structures. The grating generated in the study a 15x15 was fabricated on silicon nitride medium with diffraction efficiency as high as 65%. The calculated diffraction efficiency was as high as 68%.

CHAPTER TWO

THEORETICAL ANALYSIS

This chapter carefully examines and analyses the placement of evolvable on-line elements such as spatial light modulators which contain independent information as well as the methodology for handling multiple objective functions. The analysis carried out on the evoluable on-line elements, was found to exhaust the independent information that is injectable into the system.

2.1 Spatial Light Modulator Fourier Analysis

In this analysis, I consider three different SLM based optical interconnection architectures and where in such architectures evolvable elements should be placed to achieve the desired result. Each architecture uses either space-invariant or a space-variant point spread function. Each architecture, in my view, is potentially valuable. The first architecture employs a single SLM which is inserted in the optical system's Fourier plane as shown in figure 2.1. The output of the written pattern on the SLM is observed on the image plane using space-invariant optical interconnection methods. For example, butterfly interconnections can be viewed as space invariant as shown in figure 2.1A. The observed output pattern is, butterfly shaped and assymetric. For this geometry, coherence requirements are neither necessary nor harmful. If the individual beams on the left are mutually incoherent, it does not matter as no beam interference is required. Should the beams be mutually coherent, the interference will simply re-distribute light within the focussed point. This approach we must state, allows the loss of one third of light in the system. I show this as a representative space-invariant case.

The second and third architectures as shown in figures 2.2 and 2.3 respectively, employ two spatial light modulators. In the case of figure 2.2, one SLM is placed in the fourier plane while the other is inserted in the image or conjugate plane of the first SLM. The resulting output from this architecture is observed on the fourier plane and is space variant. This geometry provides quite an interesting task which we call Global Mapping. It can convert one arbitrary 2-dimensional pattern into another. The possibility of this happening, squarely depends on the input pattern being coherently illuminated and the insertion of SLMs in two conjugate planes.

The third architecture of figure 2.3, is another case of space variant optical interconnection. Two SLMs are placed in the image and fourier planes. The output is observed on the image plane. Again, this architecture absolutely requires that the illuminating beams be mutually coherent.

From this study, it has been found that the two SLM planes are quite independent if they are in Fourier conjugate. Further SLMs can insert no additional information. Also, the mask written on the SLMs should be a phase-only mask and the output should be observed on the fourier plane of the last mask. In Table 2.1, a matrix summary of the above SLM analysis is presented.

2.2 **Handling Multiple Objective Functions**

The employment of an optimization technique such as Genetic Algorithms for the solutions of objective functions, has generally been restricted to single cost or objective function [12]. An example of a single objective function is as shown in equation 2.1.

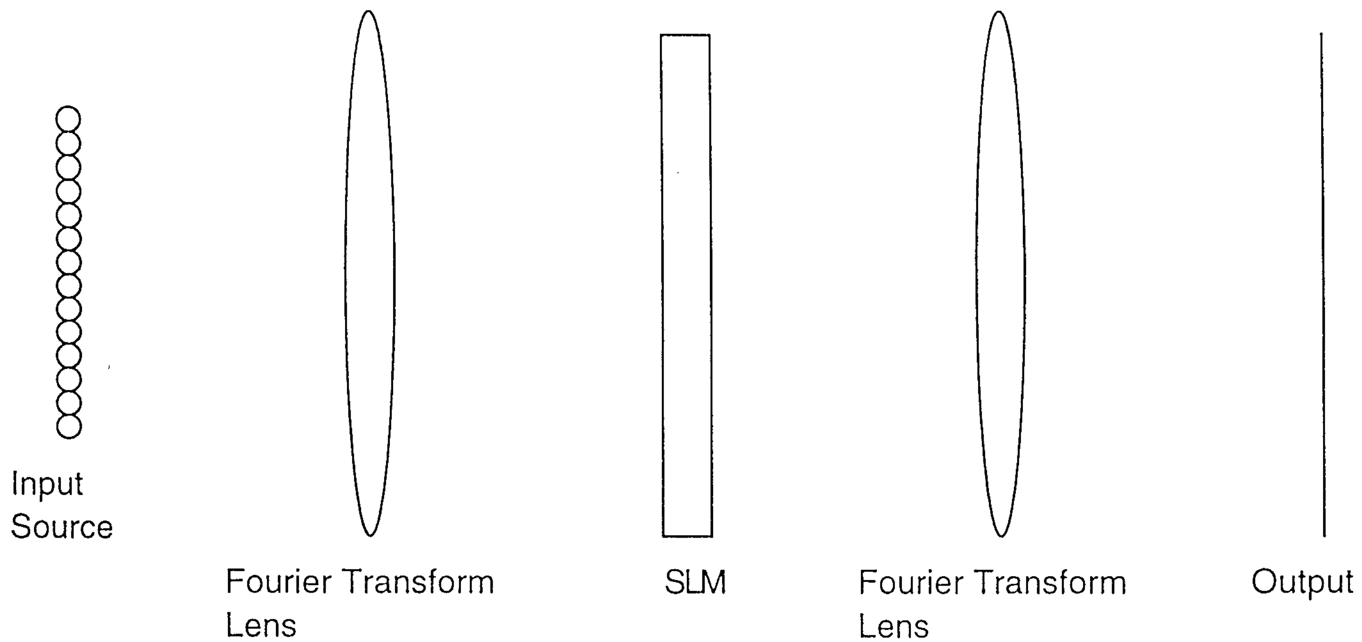
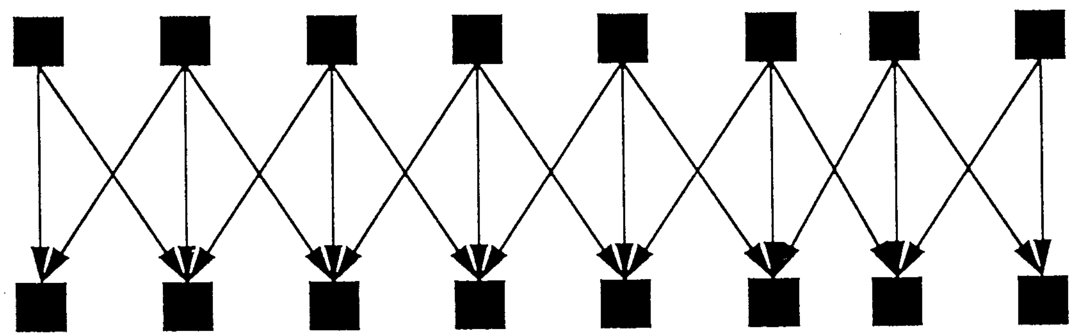


Fig: 2.1 Space invariant interconnection based on two successive optical fourier transform lenses with SLM in the transform plane.



2.1A Butterfly Space-invariant Optical Interconnection.

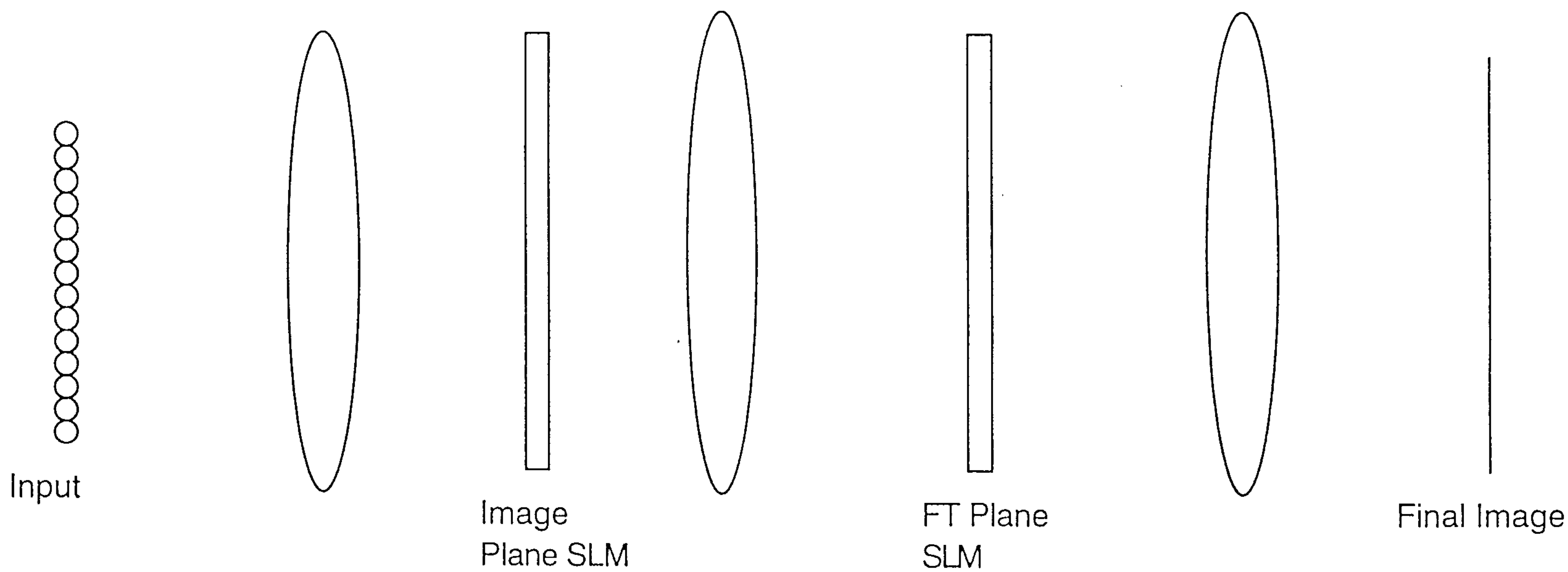
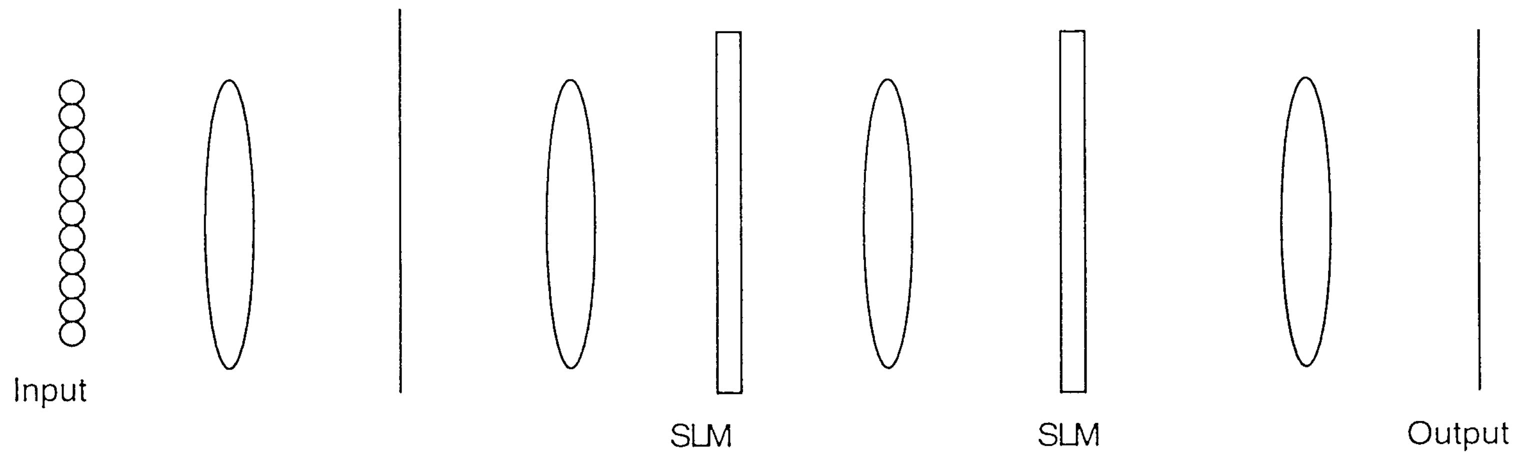


Fig 2.2 A coherent global transform architecture with two SLMs in the image and transform planes.



2.3 Coherent global transform architecture with SLMs in the image and fourier transform planes.

	SLM LOCATION		OBSERVATION PLANE LOCATION
	1	2	
Space Invariant	FOURIER		IMAGE
Space Variant 1	FOURIER	IMAGE	FOURIER
Space Variant 11	IMAGE	FOURIER	IMAGE
Space Invariant 11	IMAGE		FOURIER

Table 2.1. Matrix presentation of SLM Fourier Analysis

$$f = \sum_{i=1}^{N} x_i^2 \tag{2.1}$$

where $f(x)$ is the objective function which is to be maximised or minimised in a search space. But in this work, and for the first time, we show how multiple objective functions can be handled using Genetic Algorithms. Two possible approaches are studied, but do not exhaust the possibilities. They are a good place to start. The first of the two possible ways we call the combined figure of merit approach. Both additive and multiplicative versions follow. The additive method uses a linear combination of the figures of merit.

For two figures of merit M1 and M2,

$$S = \alpha M1 + (1-\alpha)M2, \tag{2.2}$$

for $0 \le \alpha \le 1$.

The multiplicative method is given below as

$$P = M1^{\alpha} M2^{(1-\alpha)} . \tag{2.3}$$

Without loss of generality, we have assumed M1 and M2 are to be jointly maximized. Considering equations 2.2 and 2.3 respectively, it is clear that for $\alpha = 1$, S = P = M_1, and $\alpha = 0$, S = P = M_2. The intermediate cases are especially interesting.

The second approach is to use M1 and M2 independently and sequentially. A stochastic experiment can be set up which chooses to optimize

M2 or (log M2) a fraction of (1- α) of the generations and to maximize M1 or (log M1) a fraction α of the generations.

CHAPTER THREE

DESIGN AND EXPERIMENTAL OVERVIEW

This chapter presents the process involved in the design and fabrication of a binary optical element proposed for this study. First, the process involved the utilisation of an optimization algorithm designed to find the best solution for the optical element. Second, the optimized computer generated gratings are then fabricated on a photolithographic film as an amplitude mask. Third, the fabricated optical element could then be optically evaluated using a conventional "4f" Fourier transform (FT) lens systems shown in figure 3.1

3.1 **Computer Simulation**

The generation or production of the particular binary optical element proposed for this research was designed and fabricated by us. The generation of the grating structures were accomplished by the use of the evolutionary strategies of Genetic Algorithms: a nonlinear optimization technique such as the well known Simulated Annealing [26].

To employ the use of Genetic Algorithms, we imported into our computer system Genetic Algorithm software packages which are available in public domain. The software packages include the well known Genesis 5.0 version designed by John Grefenstette of the Navy Center for Applied Research in Artificial Intelligence and GenesYs 1.0 package designed by Thomas Back of the University of Dortmund. These packages were designed for general applications of Genetic Algorithms. I used Genesis 5.0 only to generate a 9x9 grating with the application of standard mutation and one point crossover scheme. The computation of this size grating is less tasking on the

Sun workstation at Rome Laboratory. GenesYs 1.0 package was also used to generate a 9x9 grating structure. In addition, the package was used to generate 17x17, 33x33 and 65x65 size Dammann Gratings with one and two point crossover scheme, standard and adaptive mutation mechanisms. GenesYs 1.0 has many advantages over Genesis 5.0, hence, we ported it into our system. These many advantages found in this package enabled me to carry out the difficult task in the computations of the larger size Dammann Gratings. The advantages over Genesis 5.0 or features implemented in GenesYs 1.0 include proportional selection, linear ranking, Whitley's linear ranking, uniform ranking, uniform ranking with copying, inverse linear ranking, and m-point crossover.

Having successfully imported the two packages in our system, a novel objective function defined by us as described in chapter two, as well as an optional population initialization function, was used in the simulation and generation of the binary element. The implemented objective function, is the linear combination of diffraction efficiency and reconstruction error (maximum relative deviation of intensity from the ideal intensity) [20]. With this chosen objective function, we designed a computer program that implements the function and hence ported into the main Genetic Algorithm code. This program is listed in Appendix A together with a Fast Fourier Transform code obtained from numerical Recipes in "C" [19]. Then mapping of each gene to a two level binary phase mask i.e. 0 and π was carried out. A one bit corresponds to a pixel of positive amplitude (i.e. +1), and each zero bit corresponds to a negative amplitude (i.e. -1). Each calculated binary string holds a number of bit elements corresponding to the size of the Dammann Grating. For example, a 9x9 grating requires 256 bit elements, 17x17 a 512 bit element, 33x33 a 1024 bit element while a 65x65 grating holds 2048 elements. A Fast Fourier Transform (FFT) of this binary input mask was then calculated. Given that the input pattern is real and symmetric, the output is therefore real and symmetric by the properties of the fourier transform [27]. For this reason, the Discrete Cosine Transform (DCT) was employed which is a special form of the FFT which is four times more efficient since it exploits the two symmetries noted above. The DCT algorithm of the transform of the input mask is also listed in Appendix A.

For Genetic Algorithms to find solutions to a given task, it requires a population of candidate solutions which are made up of ones and zeros. For example:

10001010100011101

00111101110110101

11010101110011010

00010101100111010

To this end, the population of candidate solutions were randomly generated in our system and was used in the simulation process of the binary elements. And by using the Sun workstation, I then proceeded to simulate and optimize the binary optical elements by applying various genetic algorithms parameters such as crossover and mutation.

3.2 Fabrication of Amplitude Dammann Grating

With acceptable convergence result from the simulation of the grating, I proceeded to fabricate an amplitude grating of the various array sizes generated for this work. The process involved using a hand coded Postscript program we had designed. This program takes the mask pattern with a defined resolution and makes repeated images of the grating cells. After accomplishing this task, we sent the images on a floppy disk to an imagesetting service bureau to transfer onto a lithographic film at a resolution of 2540 dpi (dots per inch).

3.3 Optical Evaluation of Amplitude Grating

Using the optical processor shown in figure 3.1, one can evaluate one or all the manufactured optical elements for this work for comparison with the computer generated results. Here as shown in the figure, a 632 nm He-Ne laser would be incident on the grating. And using a ccd camera, the array of equal light spots diffracted by the grating can be detected and hence displayed on a TV monitor connected to the camera. The signal would then be transmitted to a Spiricon Beam Analyser connected to the TV monitor. The beam analyser is used to evaluate each light spot for intensity variations as compared to the neighboring light spots.

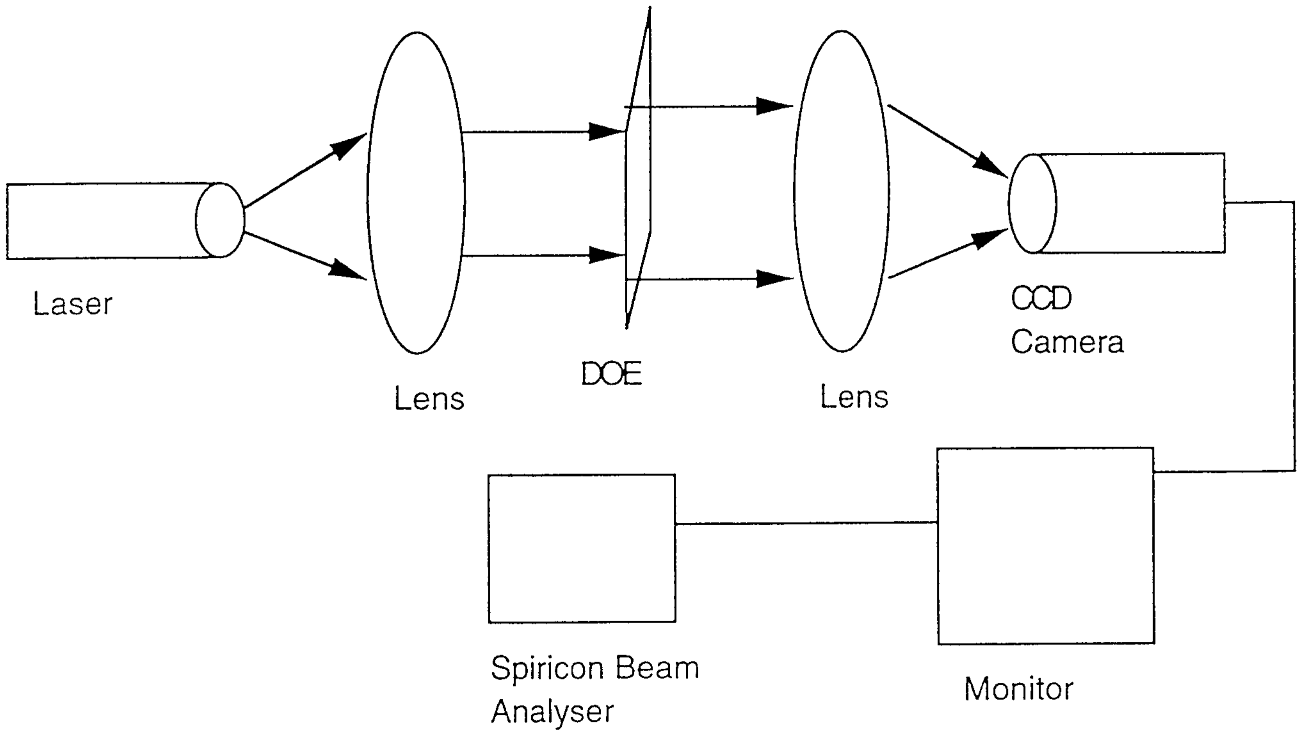


Fig. 3.1 DOE optical evaluation processor

CHAPTER FOUR

RESULTS AND DISCUSSION

The results obtained in this research work for the diffractive optical element I have devised and analysed in our laboratory are presented in this chapter. The presentation includes results from the computer simulation and fabrication of the binary optical element.

4.1 9x9 Amplitude Dammann Gratings

Employing the nonlinear optimization technique of Genetic Algorithms, I generated a 9 x 9 Dammann Gratings using our novel cost function. The cost function for this particular design, is the linear combination function as given by equation 2.2 on page 41. Genesis 5.0 Genetic Algorithm package with single crossover point by John Grefenstette was used for the simulation. After various modifications carried out on the package to suit my purpose, an objective function program was written to incorporate our cost function with the Genesis code. Here, I have utilised alpha equals to 0.5. Using different settings such as crossover rate, population size, and mutation rate, I generated various structures of a two dimensional Dammann Gratings. With a population size of 100, 256 binary elements, 0.75 crossover rate, and 0.001 mutation rate, I generated Dammann Gratings with 90% diffraction efficiency and $1.4x10^{-7}$ objective function value. The results from this simulation are shown in figures 4.1, 4.2, 4.3, and 4.4 . In a second run with population size of 200, 0.50 crossover rate, and 0.001 mutation

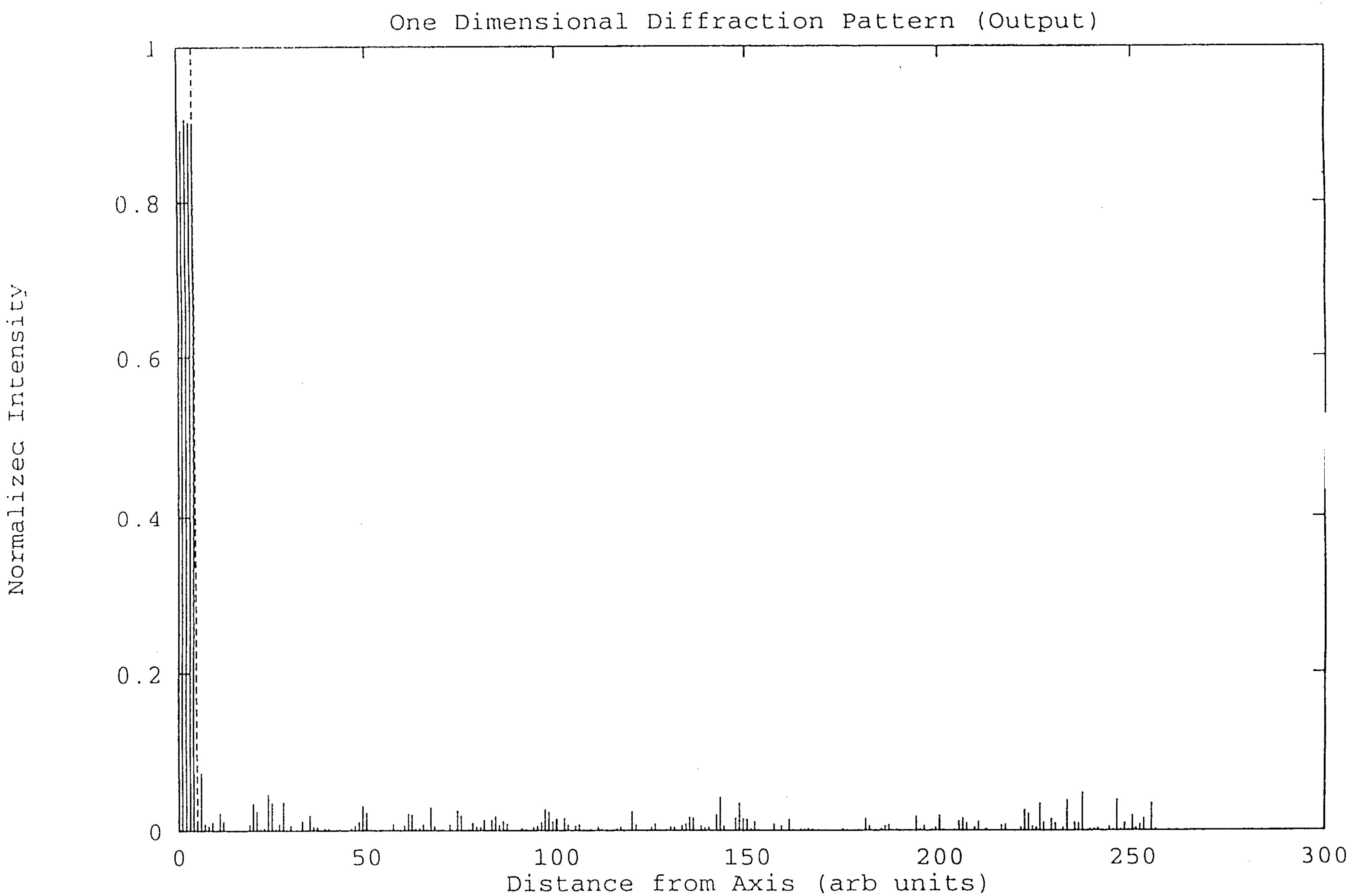


Fig. 4.1 9x9 Standard Mutation One Dimensional Diffraction Pattern

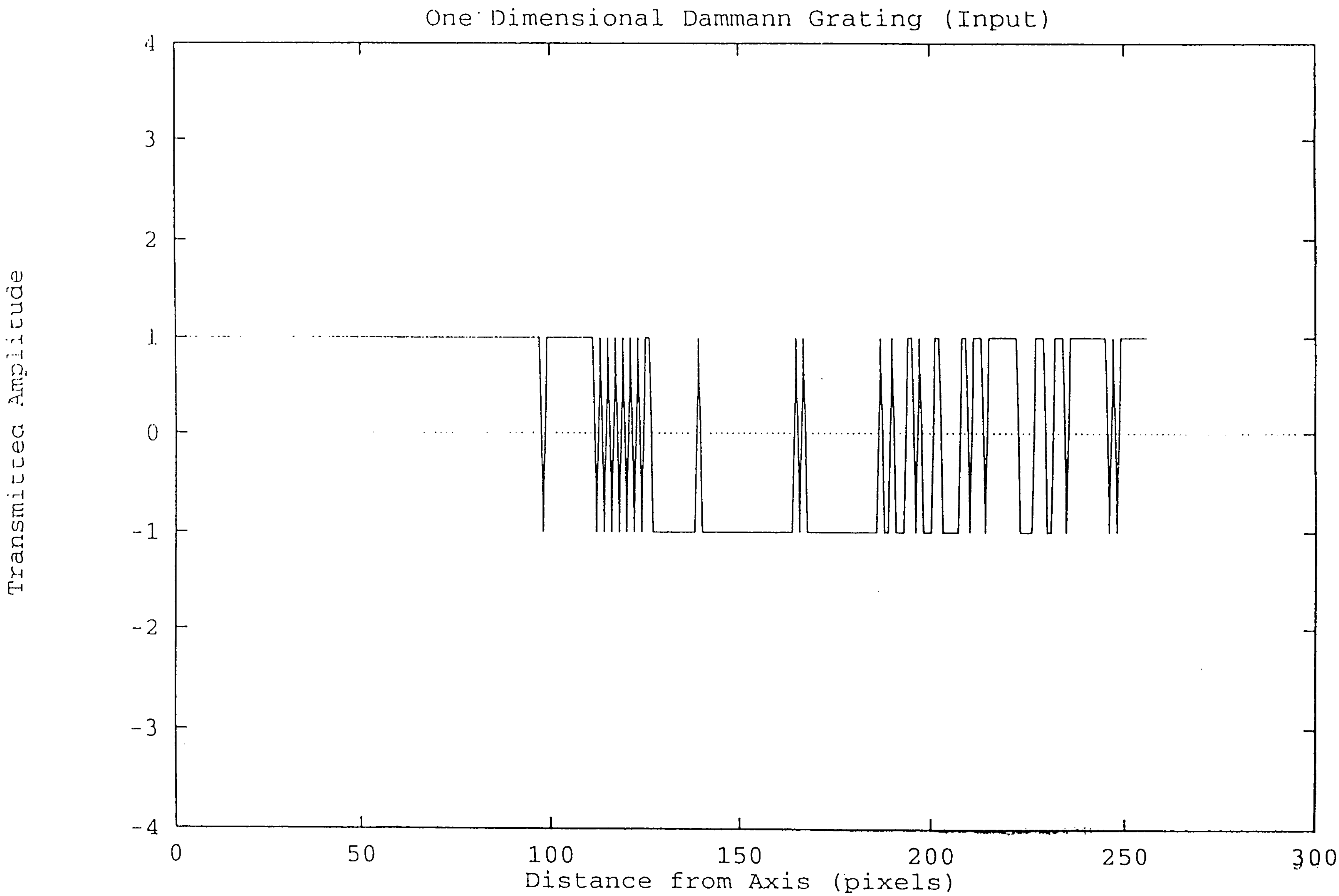


Fig. 4.2 9x9 One Dimensional Dammann Grating

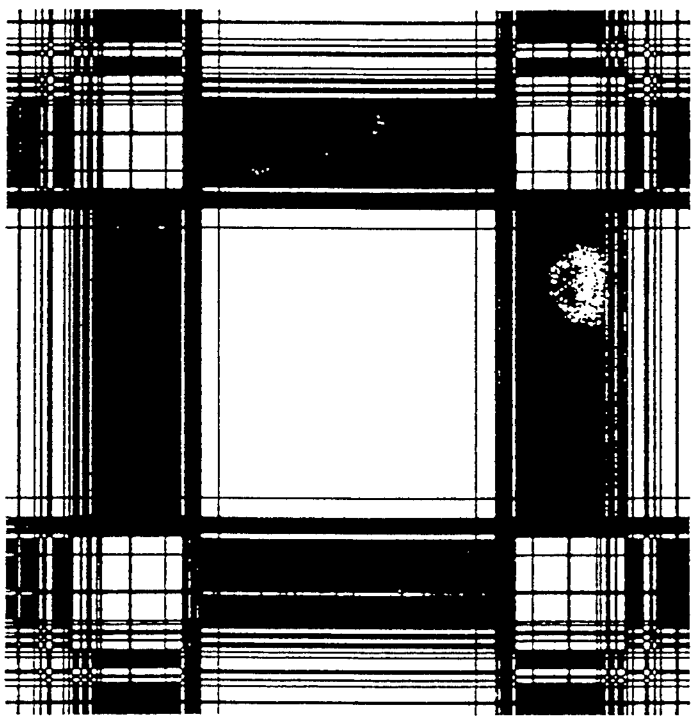


Fig. 4.3 9x9 Dammann Grating Cell

111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111
111111110101100111011101100001000000010001000001
000100000000000000010000000000001001000000010100
010000010000000001110011111011111101111101101111
1111111101011111 1.4095e-07 466 31055

Fig. 4.4 9x9 Objective Function Value

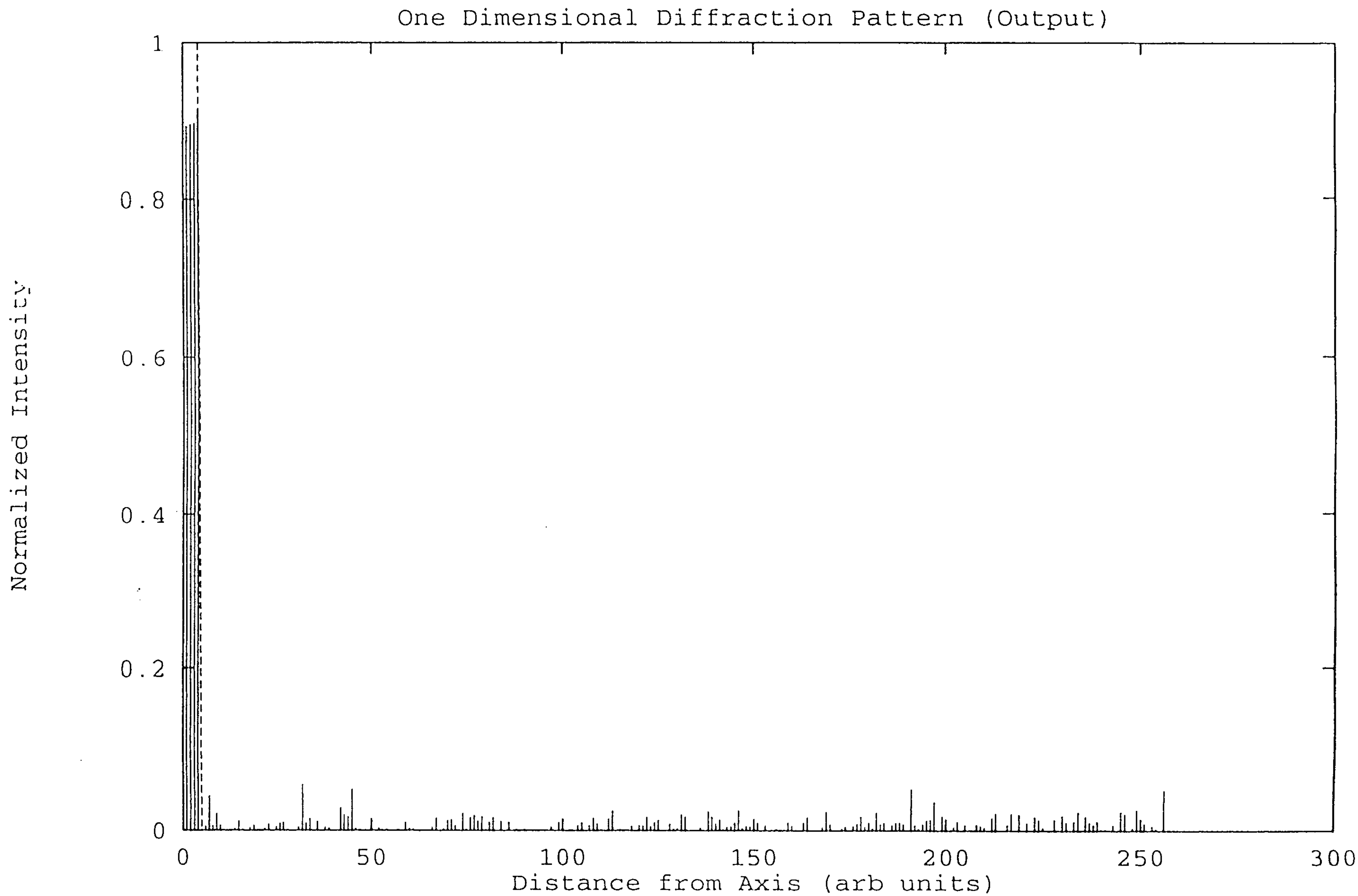


Fig. 4.5 9x9 One Dimensional Diffraction Pattern

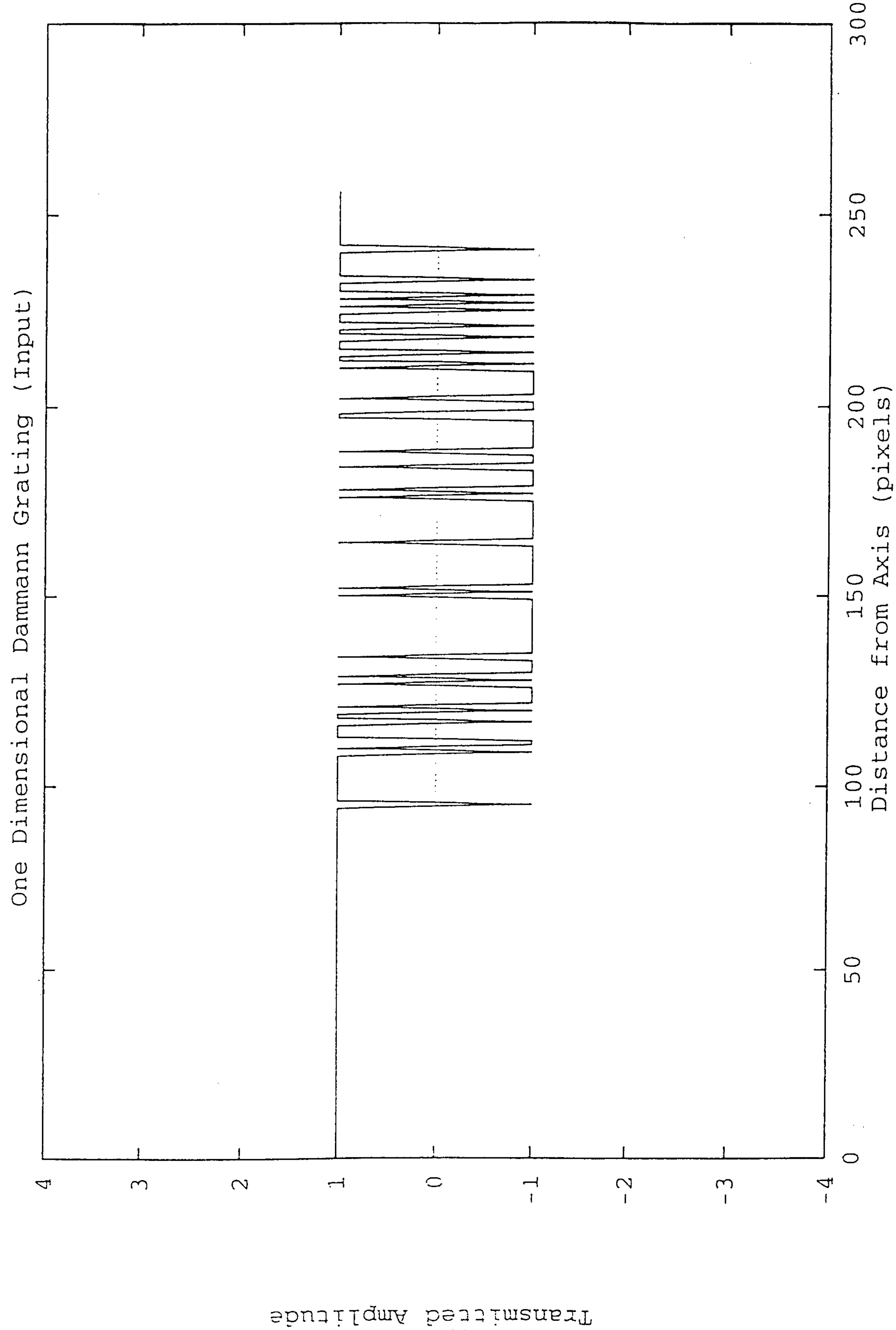


Fig. 4.6 9x9 One Dimensional Dammann Grating

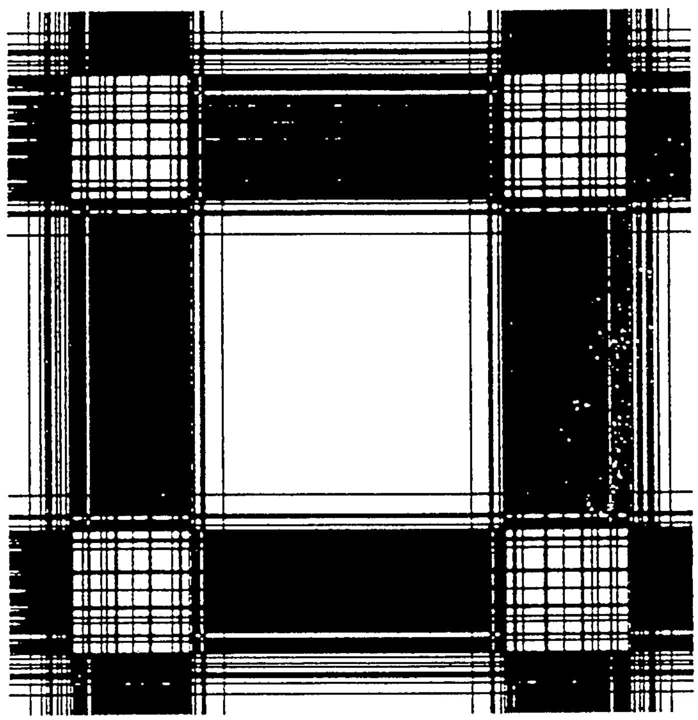


Fig. 4.7 9x9 Dammann Grating Cell

111111111111111111111111111111111111111111111111
111111111111111111111111111111111111110111111101
1111111111111111010110010001000101000100000000000
000001010000001000000000000000001000010000000010000
0100000001100101110100011111100101101101110111111
1101111111111111 4.0621e-07 310 29667

Fig. 4.8 9x9 Objective Function Value

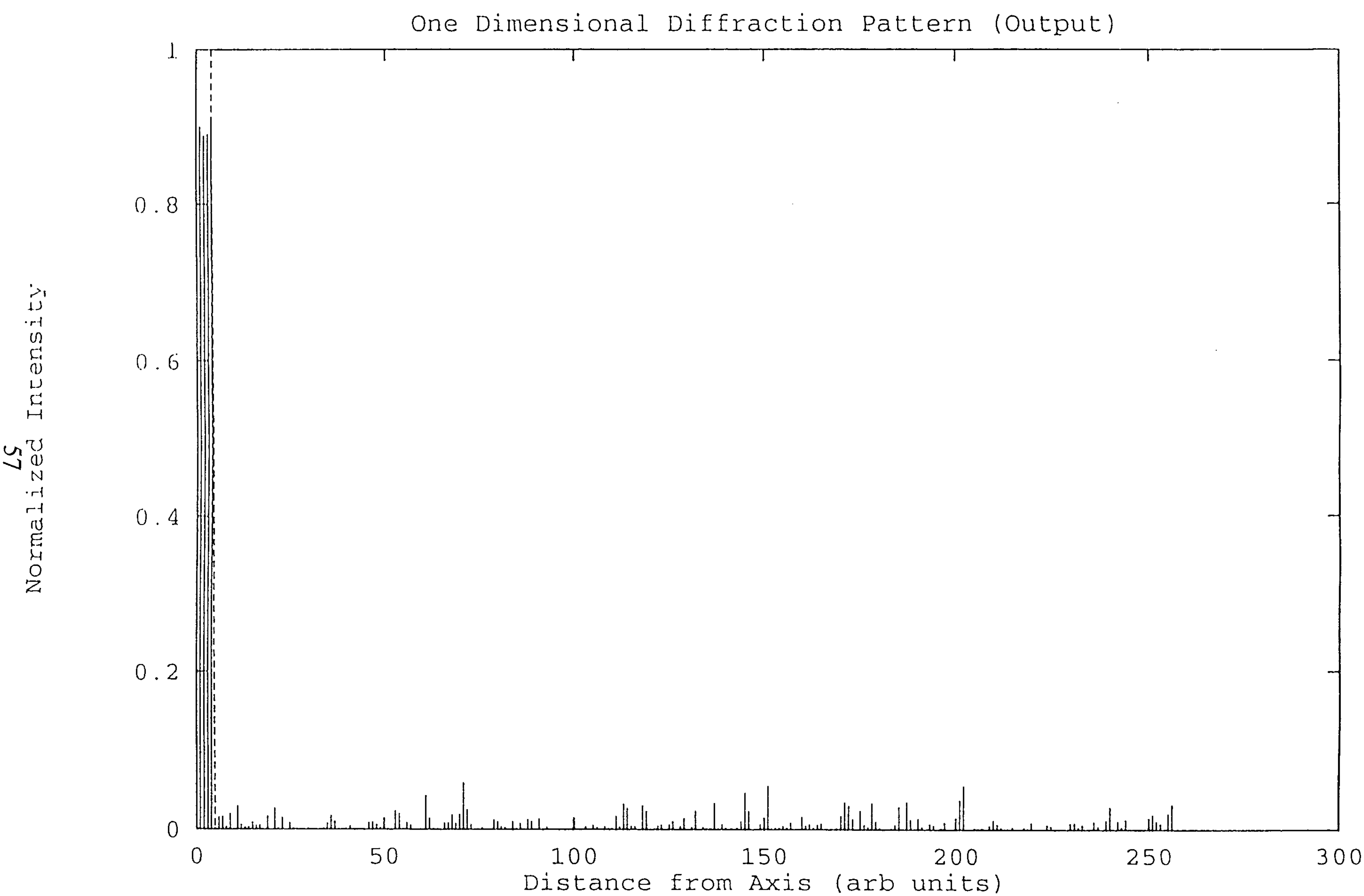


Fig. 4.9 9x9 One Dimensional Diffraction Pattern

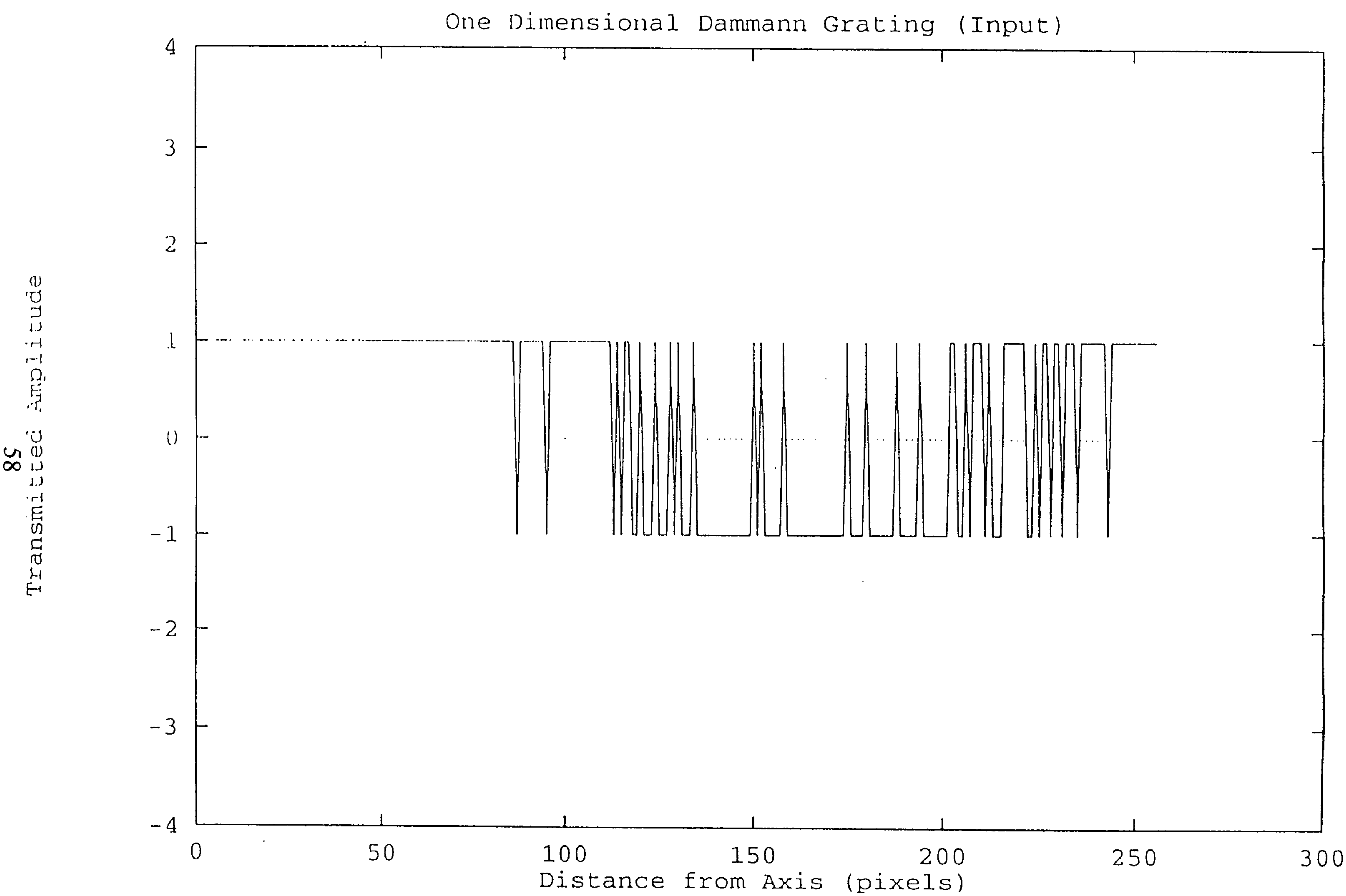
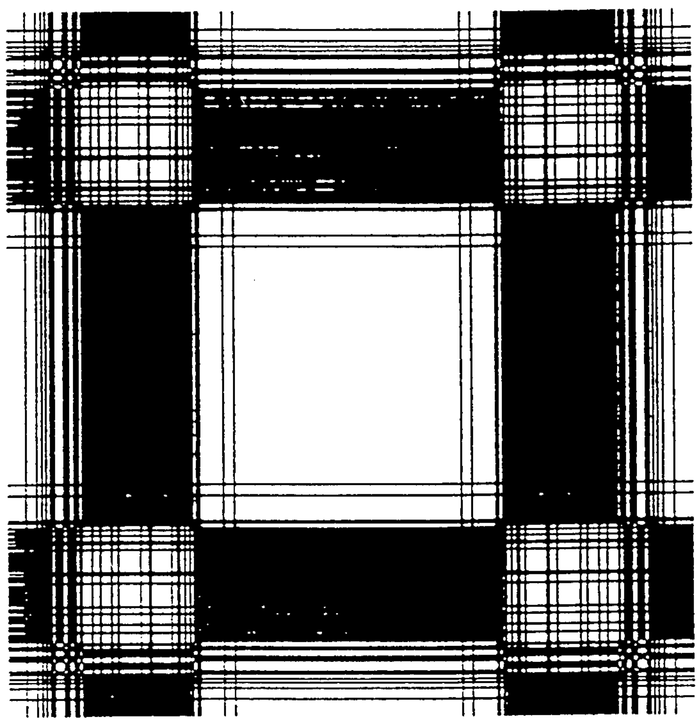


Fig. 4.10 9x9 One Dimensional Dammann Grating Structure



Dammann Grating

Fig. 4.11 9x9 Dammann Grating Cell

rate, I obtained $4.1x10^{-7}$ objective function value. The results are shown in figures 4.5, 4.6, 4.7 and 4.8 respectively. I obtained $5.7x10^{-7}$ in a third run with crossover rate of 0.25, population size of 50, and 0.001 mutation rate. The results are shown in figures 4.9, 4.10, 4.11, and 4.12 respectively. $9.1x10^{-7}$ objective function value was obtained for a run using population size of 200, 0.50 crossover rate, and 0.001 mutation rate. This run was done with "Rank based selection" option as given by the Genesis 5.0 package. The results are shown in figures 4.13, 4.14, 4.15, and 4.16 respectively. A fifth simulation with population size of 50, 0.10 mutation rate and 0.50 crossover rate with "Rank based selection" option, I obtained a value of $9.4x10^{-7}$. Figures 4.17, 4.18, 4.19, and 4.20 show the results generated from the run. These runs were done with 1500 generations lasting about fifteen to twenty minutes computing time on Sun workstation. The best result obtained from these simulations is that given by using population size of 50, 0.25 crossover rate, and 0.001 mutation rate. The use of high mutation rate and low crossover rate for simulation employing Genetic Algorithms, I found, gave undesirable results as indicated by a run using a population of 100, 0.25 crossover rate and 0.50 mutation rate. The objective function value obtained was too high, a value of 1.1x10−3. A relationship between population size, crossover rate, mutation rate and objective function is shown in figures 4.21, 4.22, and 4.23. It was observed that low objective function value is obtained if population size is within 100. At higher population sizes, the value tends to increase, which indicates the design will be undesirable. Table 4.1 shows

111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111
101111111111111010101010101011000000000000100000
000000000000000000001010000000000000000000100100
011010001100000110111011111111000011100111011111
1111101011111111 5.7267e-07 661 8926

Fig. 4.12 9x9 Objective Function Value

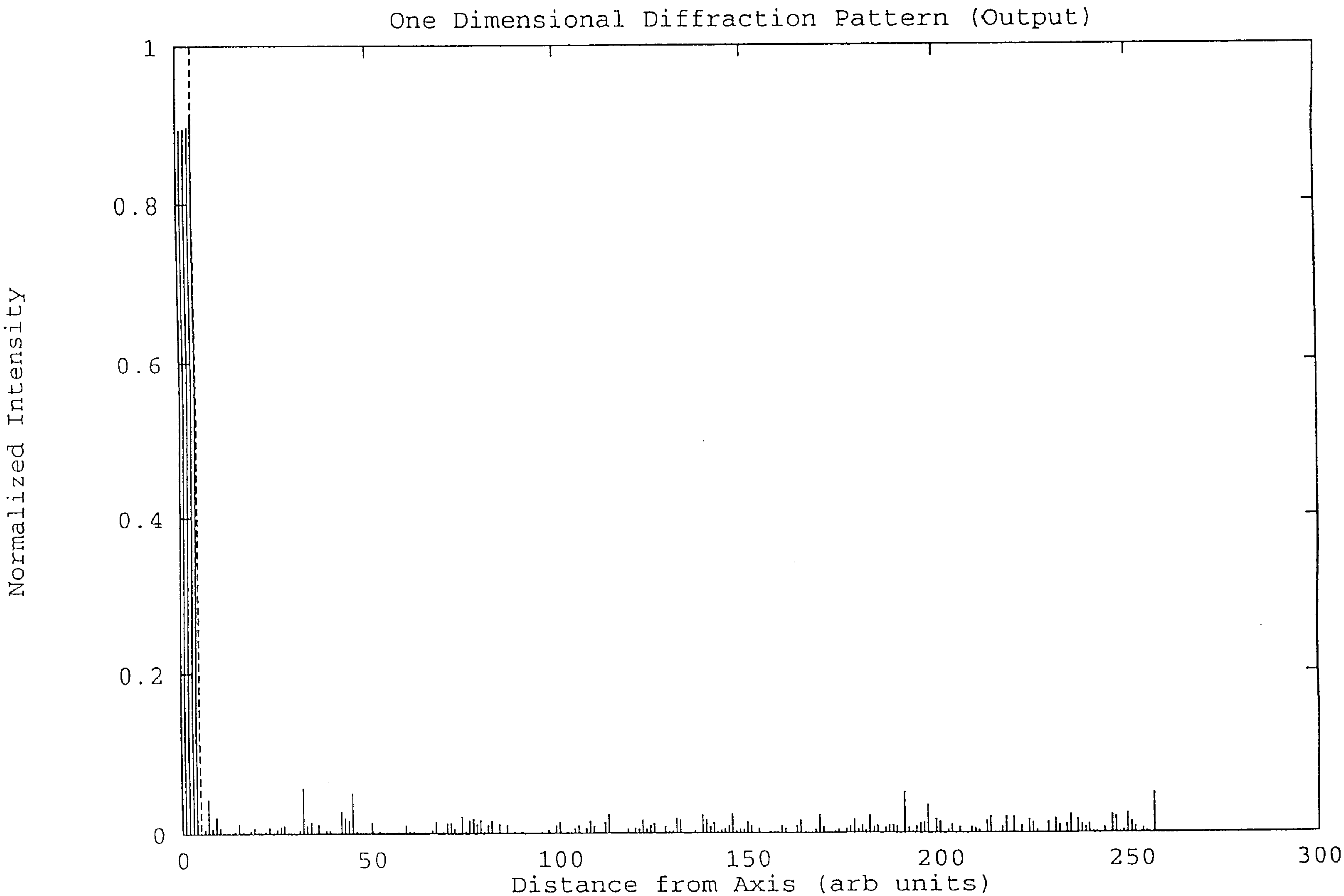


Fig. 4.13 9x9 One Dimensional Diffraction Pattern

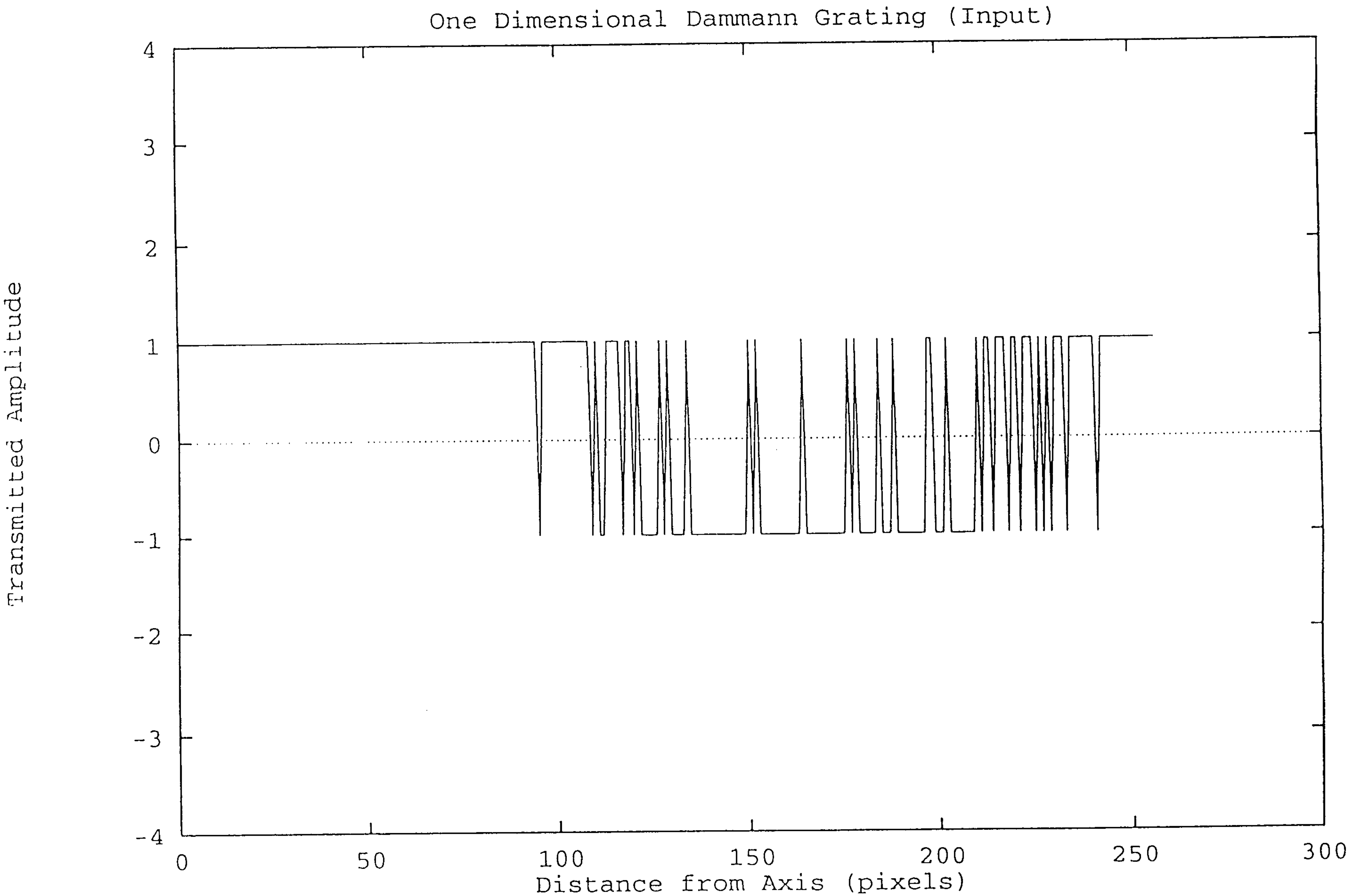
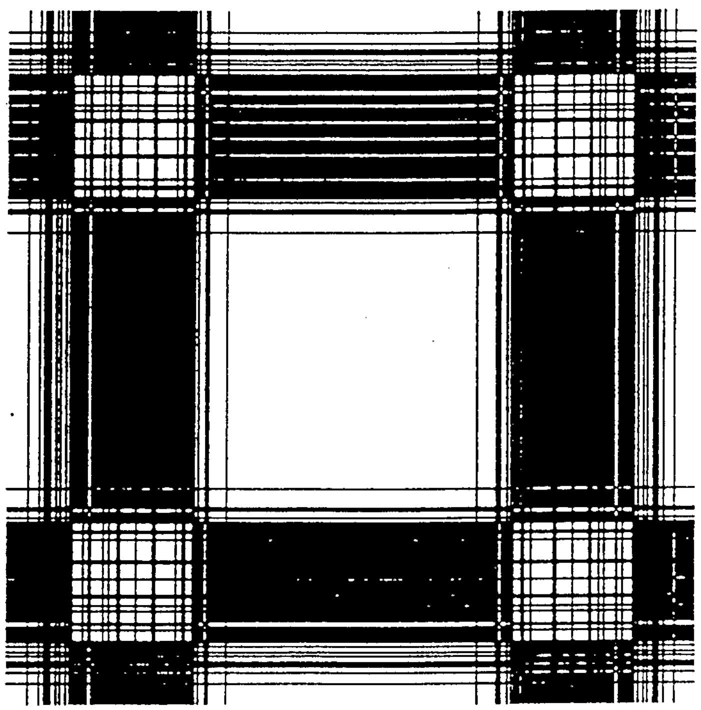


Fig. 4.14 9x9 One Dimensional Dammann Grating Structure



Dammann Grating

Fig. 4.15 9x9 Dammann Grating Cell

111111111111111111111111111111111111111111111111
1111111111111111111111111111111111111111111111101
1111111111110100111011010000010100001000000000
00000101000000000001000000000001010000001000 10000
0000110001000000010110111011011101010111011111111
0111111111111111 9.1993e-07 301 31900

Fig. 4.16 9x9 Objective Function Value

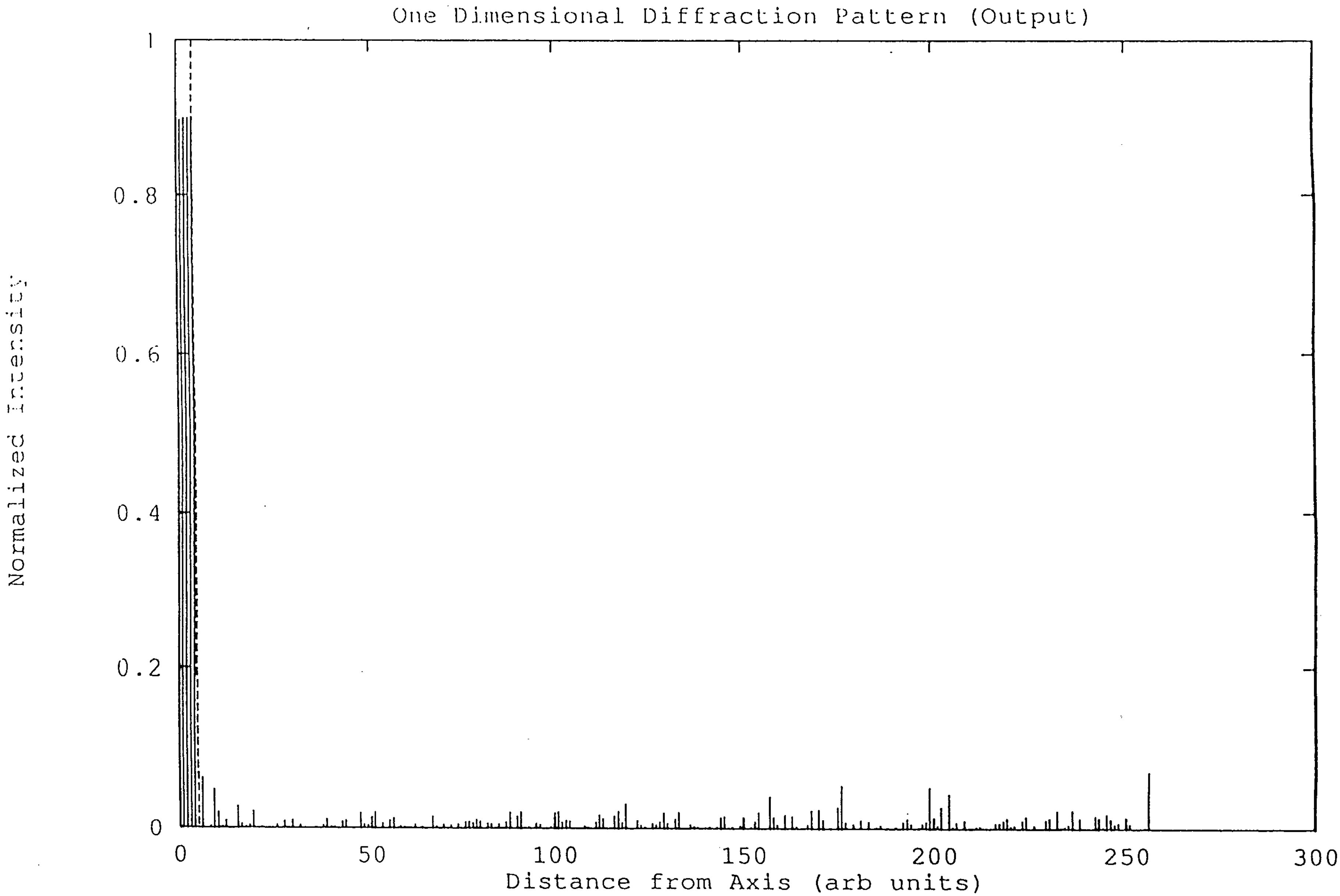


Fig. 4.17 9x9 One Dimensional Diffraction Pattern

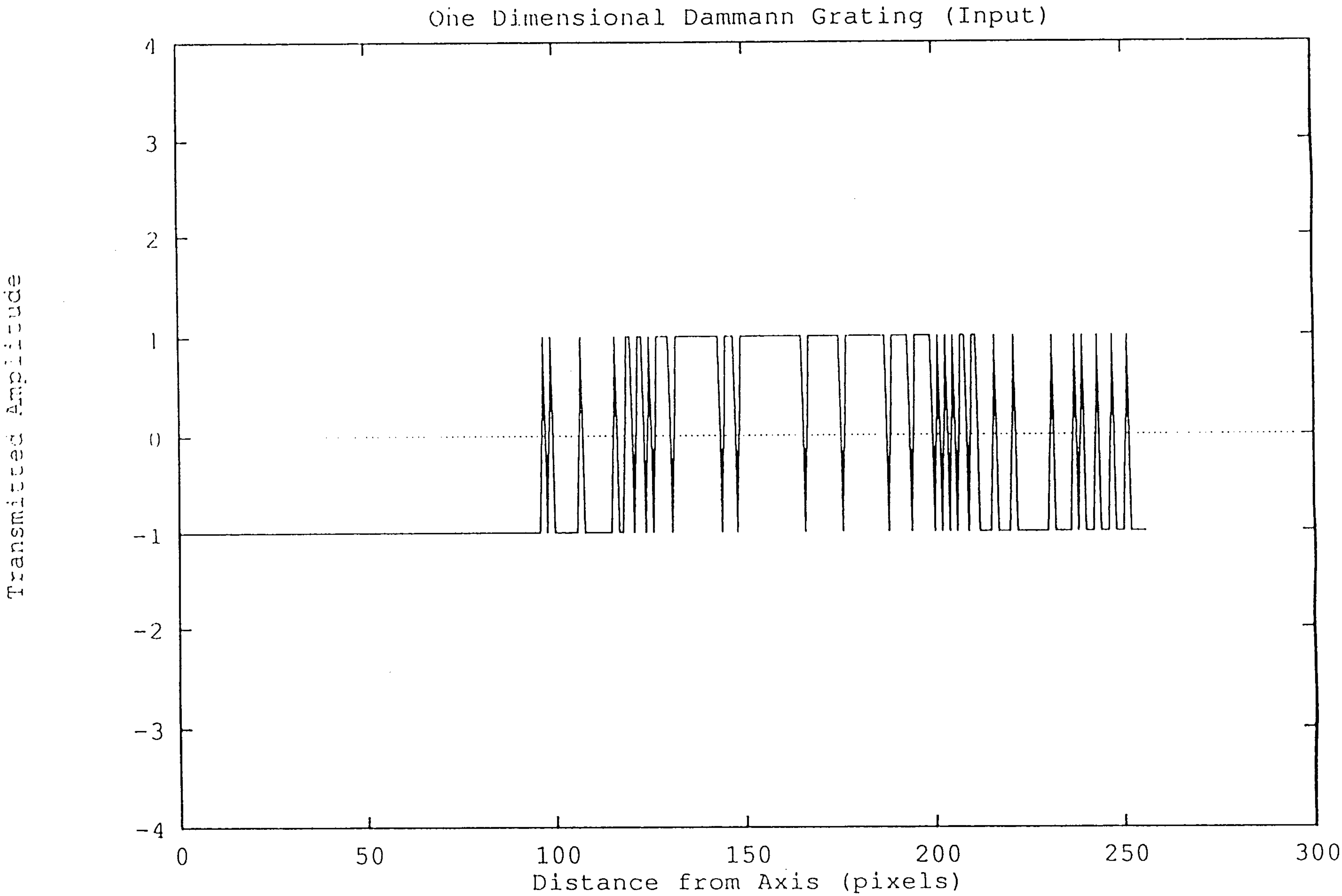
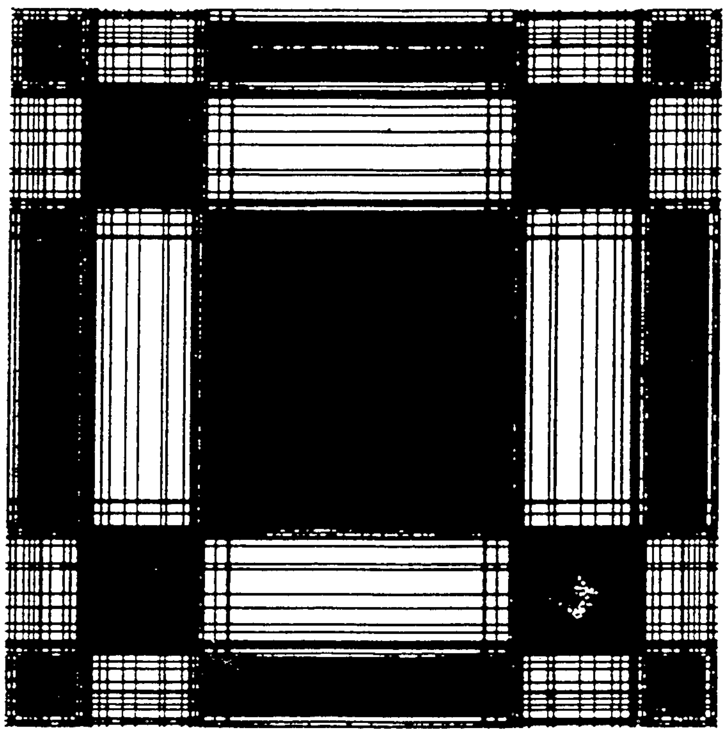


Fig. 4.18 9x9 One Dimensional Dammann Grating Structure



9x9 Dammann Grating

Fig. 4.19 9x9 Dammann Grating Cell

111111111111111111111111111111111111111111111111
111111111111111111111111111111111111111111111111
111011111101110111101000100010101000100100010000
000000000000000000000010000000000000000000010010
0001001010101111000110001111110110111111011111100
1011101111111111   9.4359e-07   753   31258

Fig. 4.20 9x9 Objective Function Value

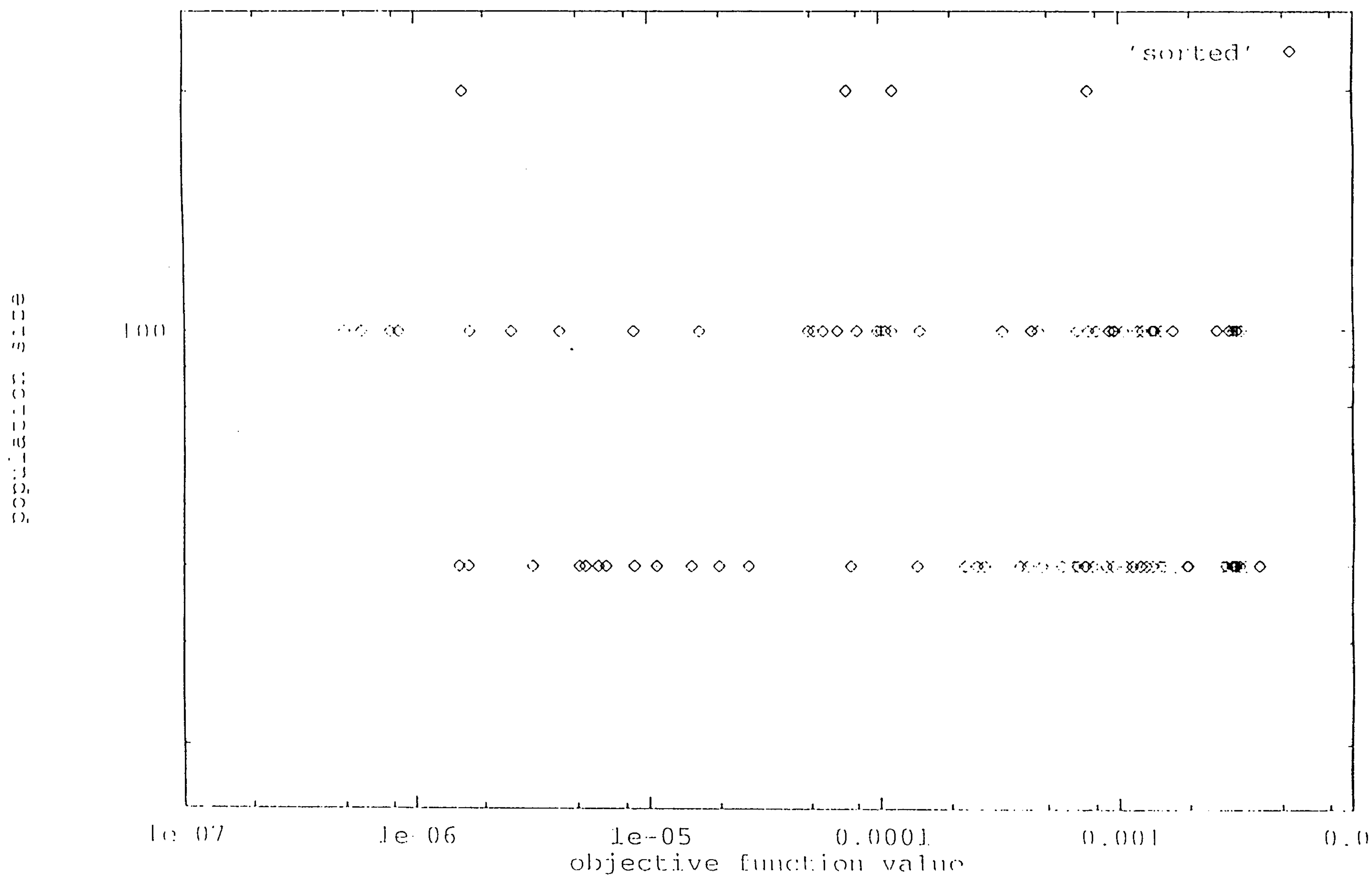


Fig. 4.21 Population Size versus Objective Function Value

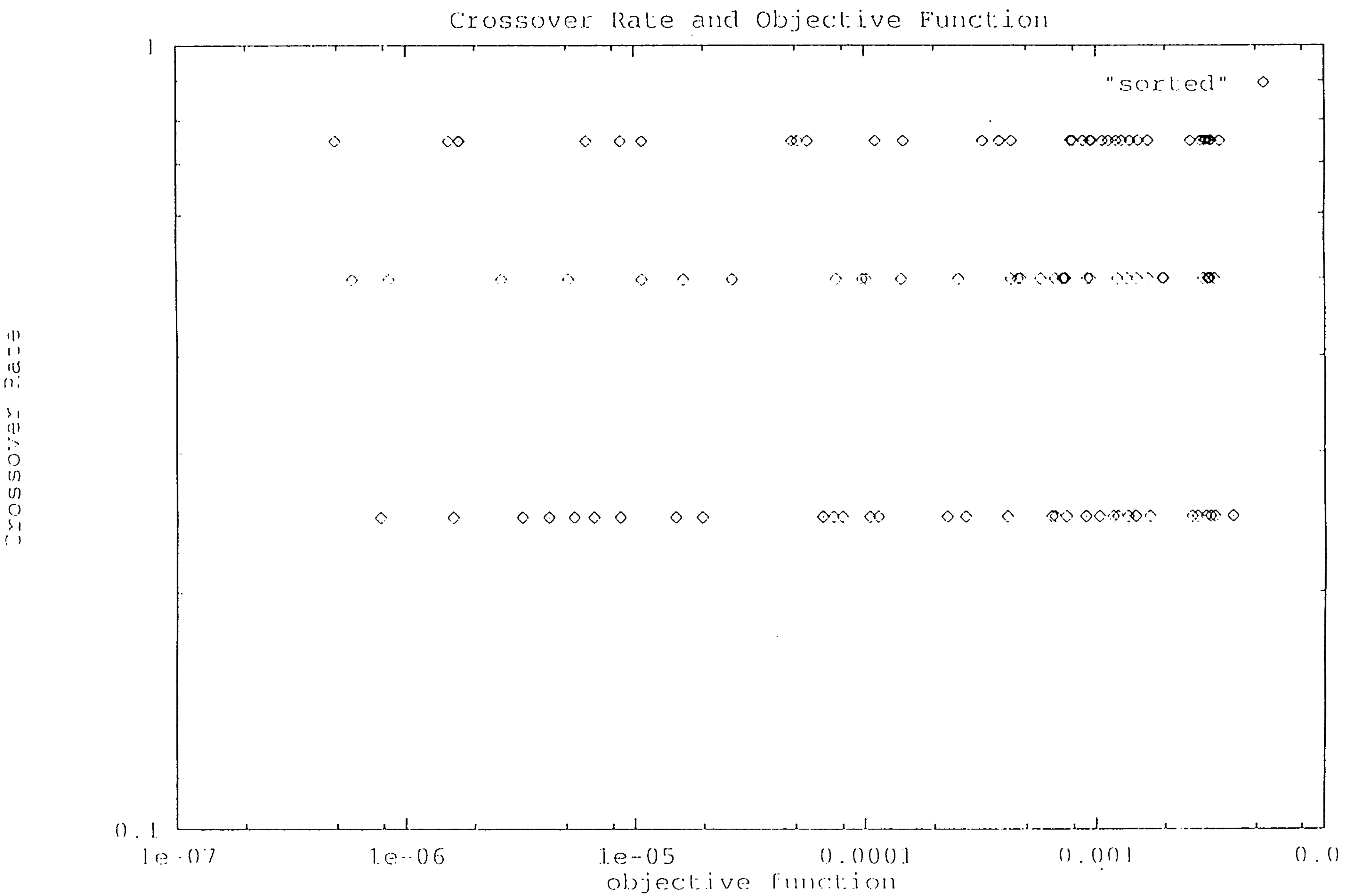


Fig. 4.22 Crossover Rate versus Objective Function Value

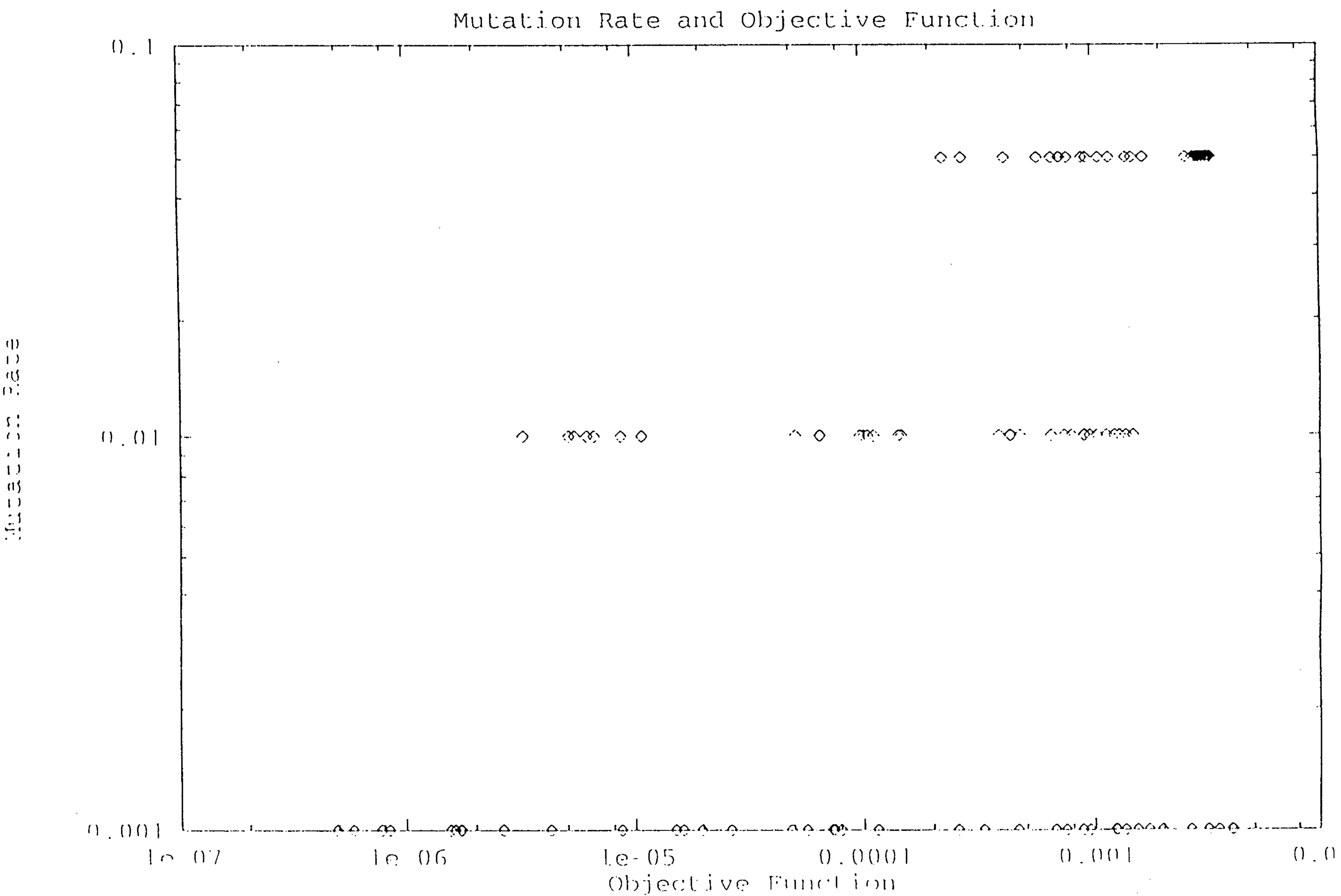


Fig. 4.23 Mutation Rate versus Objective Function Value

Table 4.1 9x9 Dammann Gratings Array With Different Setting

(Genesis 5.0)
(One Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.75	0.001	7.7×10^{-6}
100	0.75	0.001	1.4×10^{-7}
200	0.75	0.001	1.2×10^{-6}
50	0.50	0.10	7.0×10^{-6}
100	0.50	0.10	7.0×10^{-6}
200	0.50	0.10	1.6×10^{-6}
50	0.25	0.001	5.7×10^{-7}
100	0.25	0.001	2.9×10^{-6}
200	0.25	0.001	3.3×10^{-6}

various runs with different settings and objective function value for the generation of 9 x 9 Dammann Gratings.

Using the best result obtained from the simulations, I (through an imagesetting service bureau), fabricated an amplitude Dammann Gratings with 2540 dpi (dots per inch) resolution. This grating when optically evaluated will generate at Fraunhofer or the far field region, a 9x9 array of equal light spots with high diffraction efficiency. The feature sizes of this grating is about $7\mu m$. The fabricated amplitude grating is shown in figure 4.24.

Using the Genetic Algorithm code by Thomas Back (GenesYs 1.0) which was modified by us for our purpose. Employing a two point crossover scheme and standard mutation, I generated a 9x9 Dammann Grating arrays at 3000 number of generations. Also, a generation of the above size gratings was carried out using adaptive mutation and two point crossover mechanism. Simulation of this optical element was also carried out using one point crossover scheme with standard mutation and one point crossover scheme with adaptive mutation mechanism. Standard mutation scheme allows one the freedom to determine what mutation probability to use in a given generation or simulation, while adaptive mutation incorporates the mutation probability into the individual's bit string or genotype. The results obtained from applying low standard mutation rate as was observed from the simulations did not differ significantly from the adaptive mutation scheme. A major difference was however observed when high standard mutation rate was applied in the simulation compared to adaptive mutation mutation rate. It was also observed that

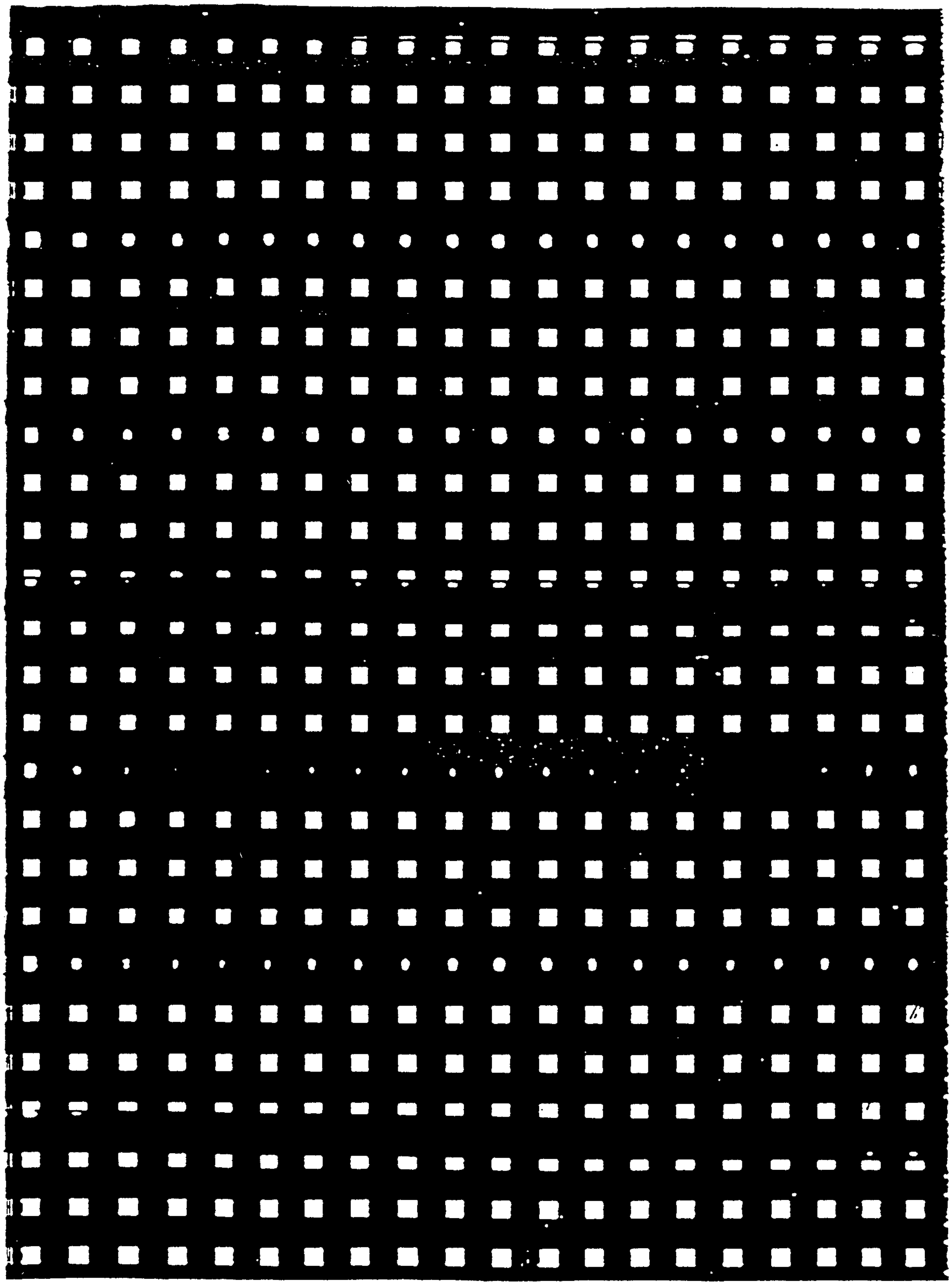


Fig. 4.24 9x9 Fabricated Amplitude Dammann Grating Device

Table 4.2 9x9 Dammann Grating Arrays with different Genetic Algorithms setting (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.75	0.001	3.0507×10^{-3}
100	0.75	0.001	3.0422×10^{-3}
200	0.75	0.001	2.9738×10^{-3}
50	0.50	0.10	4.6839×10^{-3}
100	0.50	0.10	4.7431×10^{-3}
200	0.50	0.10	4.8927×10^{-3}
50	0.60	0.001	2.9285×10^{-3}
100	0.60	0.001	2.8129×10^{-3}
200	0.60	0.001	2.9440×10^{-3}

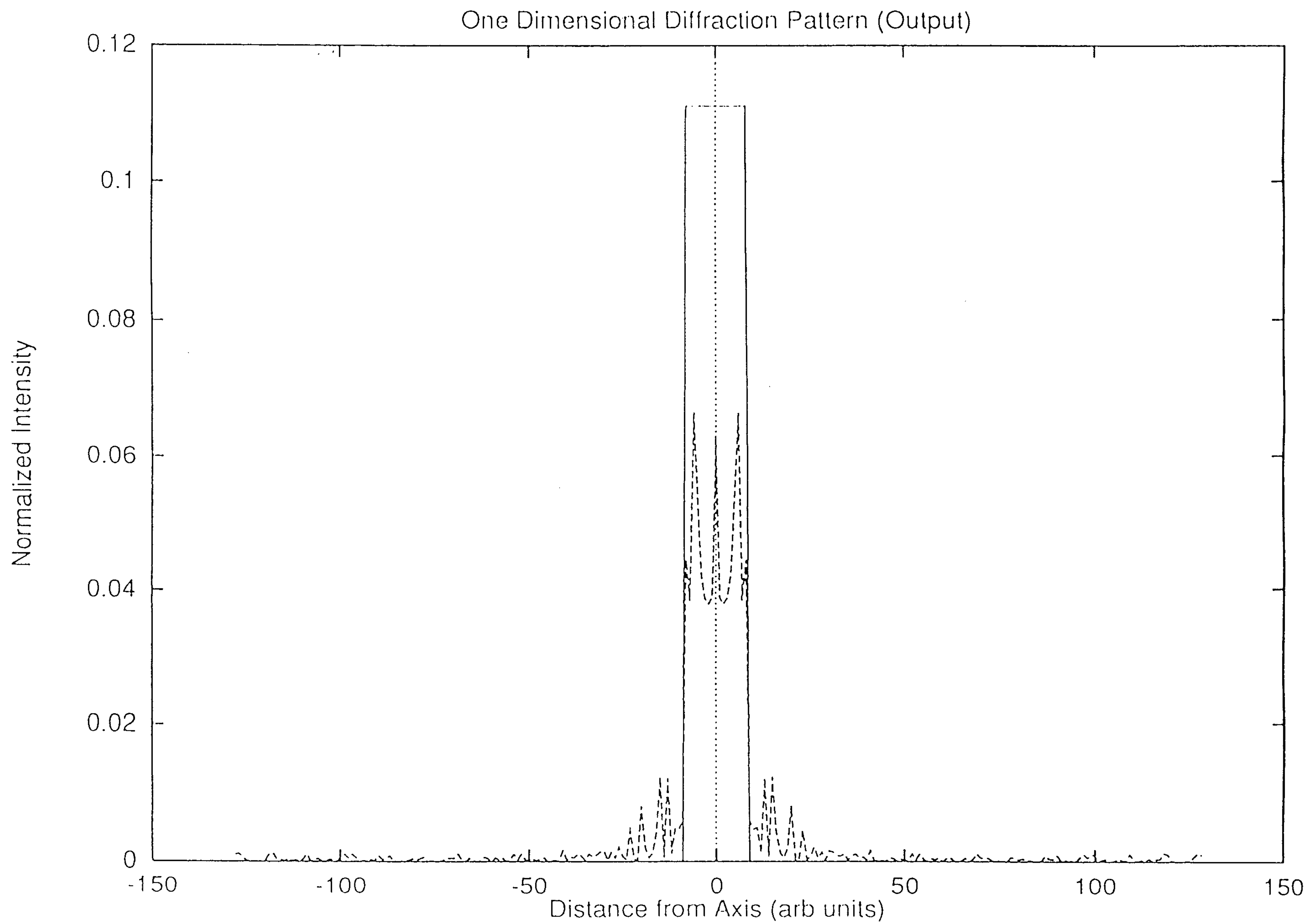


Fig. 4.25 9x9 One Dimensional Diffraction Pattern

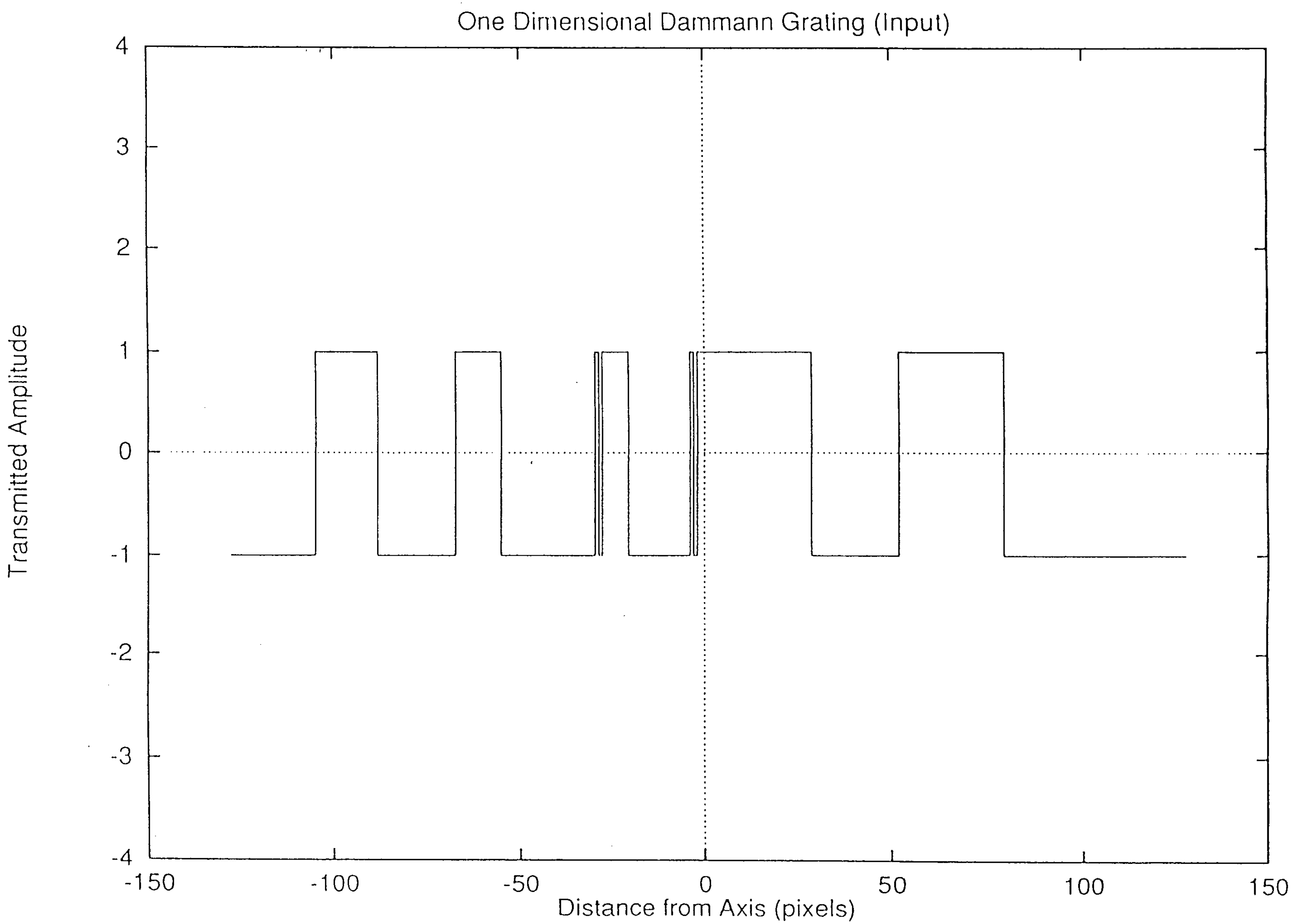


Fig. 4.26 9x9 One Dimensional Dammann Grating Structure

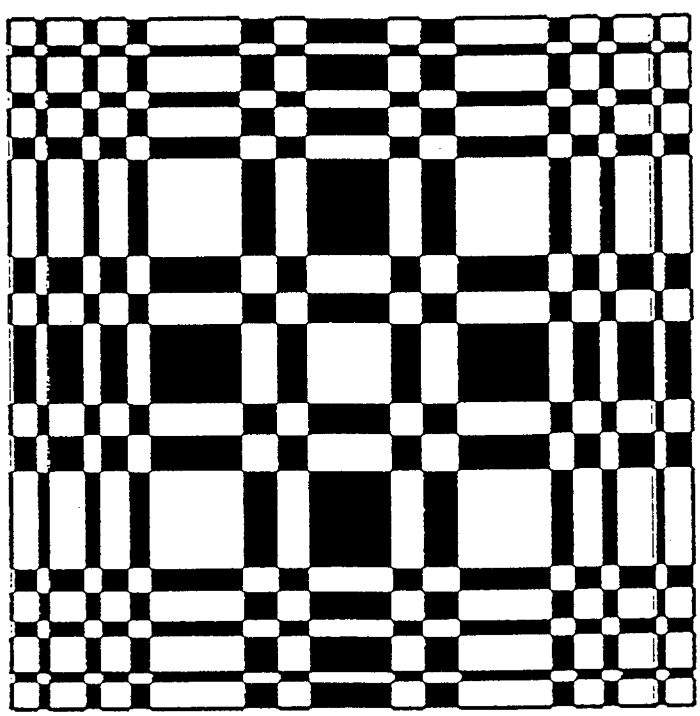


Fig. 4.27 9x9 Dammann Grating Cell

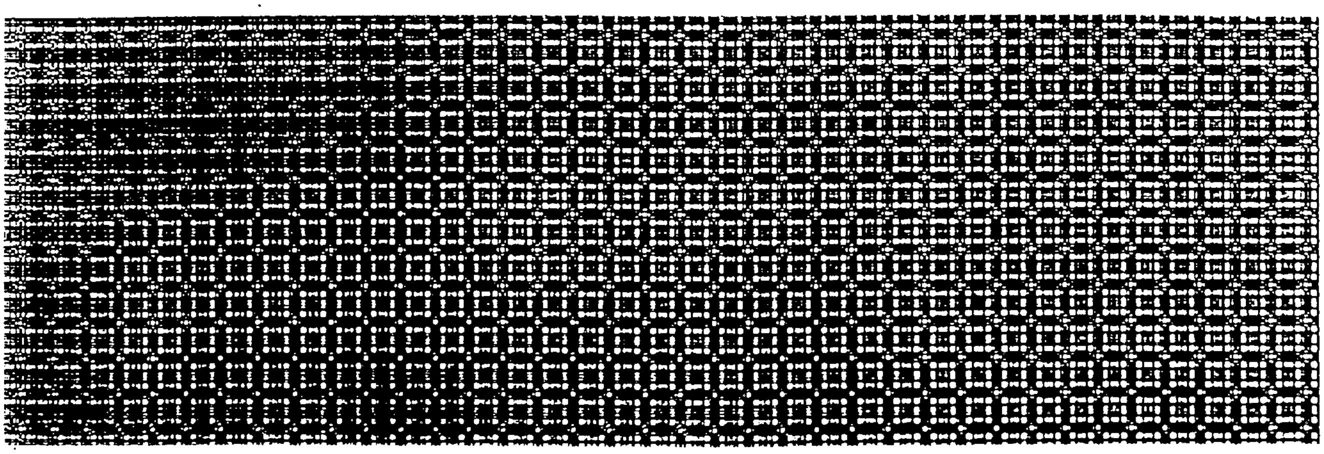

0.76 512

Fig. 4.28 9x9 Fabricated Amplitude Dammann Grating

11111111 11111111 11111111 11111000 00000000
00000000 00001111 11111111 11111111 11111110
00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000001 11111111
11111111 00000000 00000000 00000111 11111111
10000000 00000000 00000000 00101111 11100000
00000000 00001011 3.0348e-03 3207 247359

Fig. 4.29 9x9 Objective Function Value

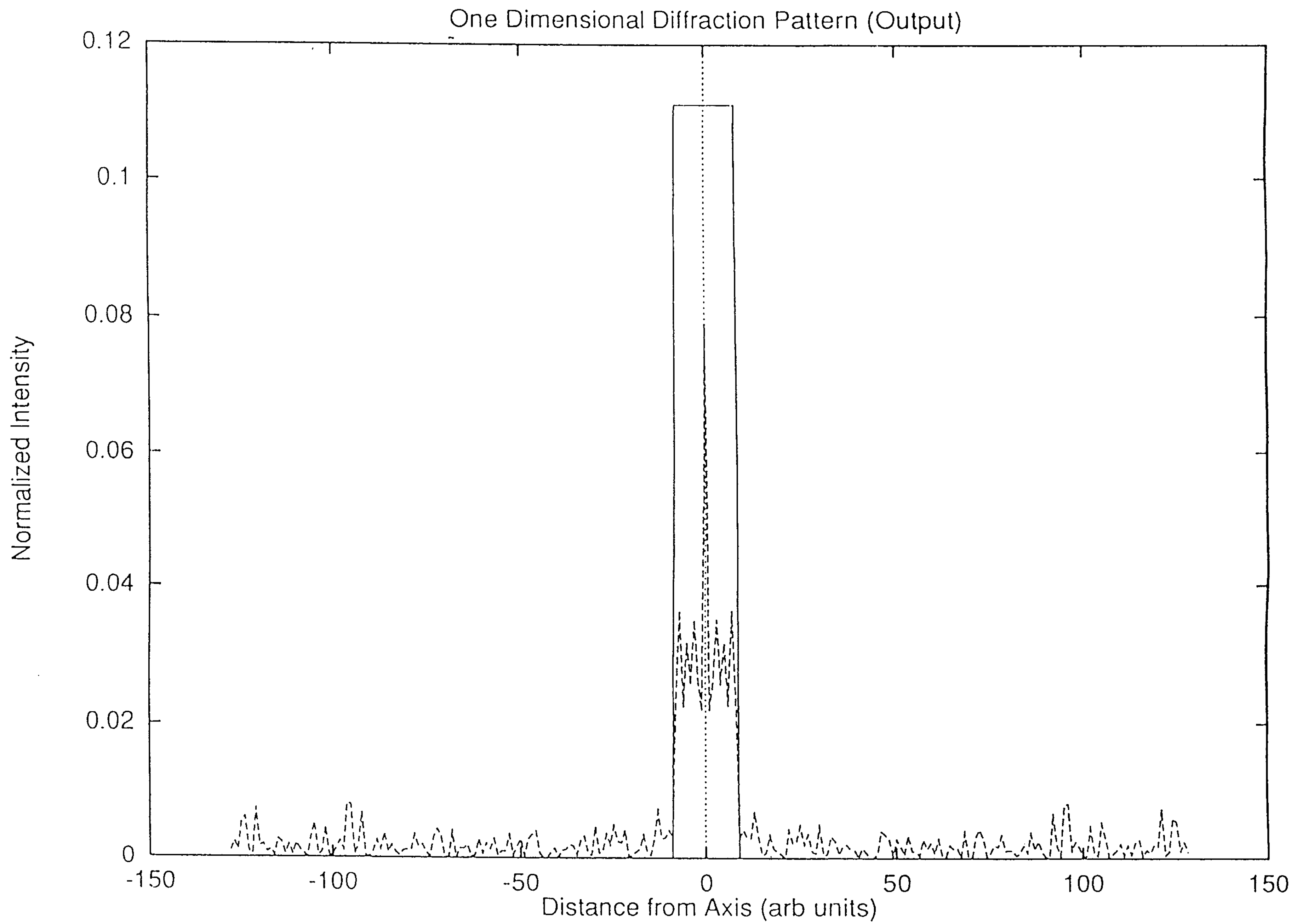


Fig. 4.30 9x9 One Dimensional Diffraction Pattern (High Mutation Rate)

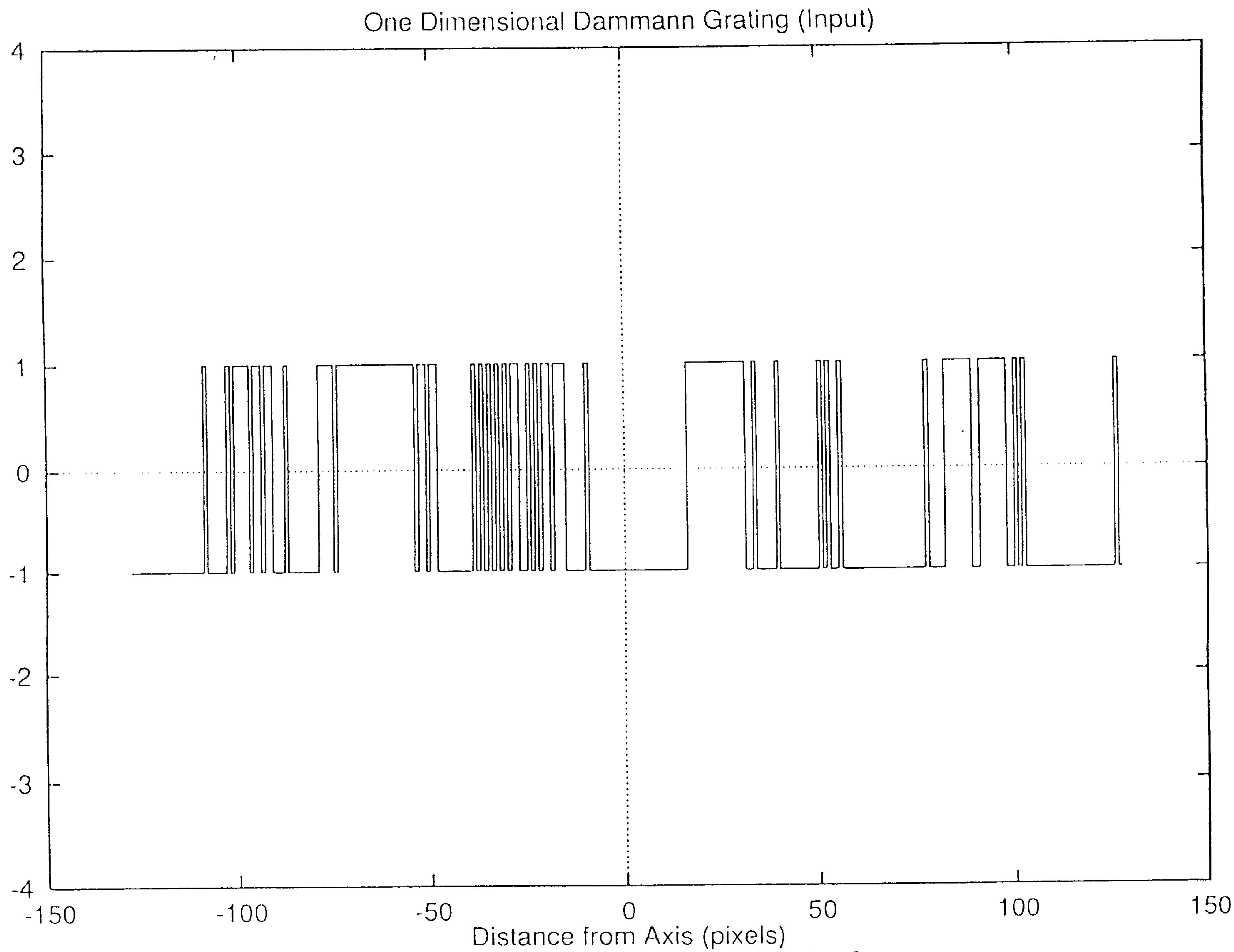
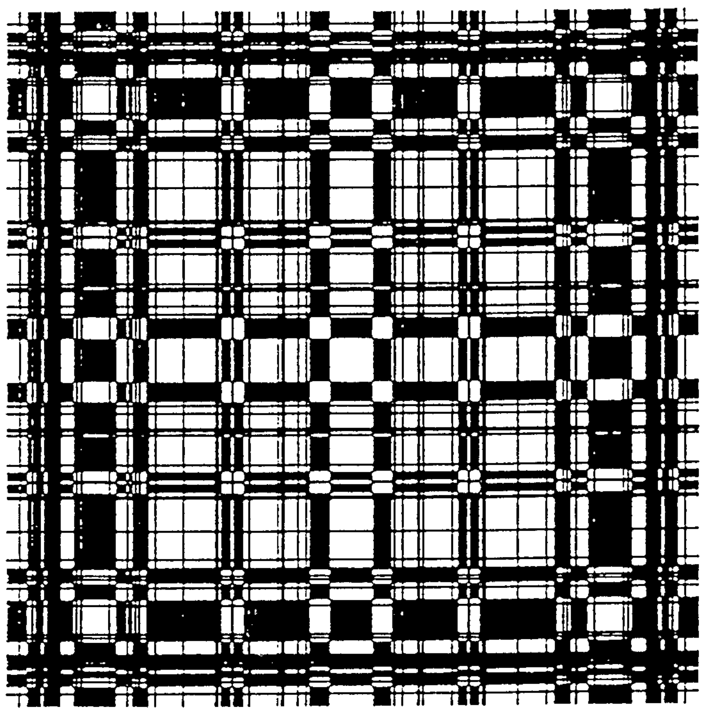


Fig. 4.31 9x9 One Dimensional High Mutation Rate Dammann Grating Structure



Dammann Grating

Fig. 4.32 9x9 Dammann Grating Cell (High Mutation Rate)

00000000 00000000 11111111 11111110 01000001
00000000 00101001 00000000 00000000 00000100
00111111 10011111 11001010 00000000 00000000
00000010 00000000 00000000 00010000 01011110
11011000 10000000 01111011 11111111 11111111
11011011 00000000 01010101 01011001 01011011
10000010 00000000 4.8776e-03 2983 298370

Fig. 4.33 9x9 Objective Function Value (High Mutation Rate)

high mutation rate produces noisy diffraction pattern and hence a grating that could not be fabricated because of very low resolution of the grating structure. Table 4.2 shows the various runs with different settings and the resulting objective function value. Figures 4.25, 4.26, 4.27, 4.28 and 4.29 show the output diffraction pattern, input Dammann Grating structure, single period two dimensional Dammann Grating cell, a fabricated 9x9 amplitude Dammann Grating and the string structure respectively with objective function value of 3.0348×10^{-3}. From figure 4.25, it is observed that the diffraction efficiency obtained for this 9x9 grating is about 50% with highly resolved diffraction pattern. The amplitude Dammann Grating fabricated on a photolithographic film shown in figure 4.28 is fabricated at a resolution of 2540 dpi (dots per inch). The input grating as shown in figure 4.26 is also highly resolved. Figures 4.30, 4.31, 4.32 and 4.33 show the results obtained with high mutation rate. As a result of the high mutation, it is observed from figure 4.30 that the diffraction efficiency is about 15%. Here, as indicated by figure 4.30, the diffraction efficiency was drastically reduced as compared to the result of figure 4.25.

The results obtained from the above simulation using adaptive mutation and two point crossover scheme are shown in table 4.3 and in figures 4.34, 4.35, 4.36, and 4.37 respectively. Again, the diffraction efficiency is about 50% with fairly resolved diffraction pattern. The input Dammann Grating as shown in figure 4.35 is also highly resolved. For the one point crossover scheme and standard mutation, the results are as shown in table 4.4 and in figures 4.38, 4.39, 4.40, and 4.41 respectively. The diffraction efficiency, again is about 50%. The

Table 4.3 9x9 Dammann Grating Arrays with different Genetic Algorithms setting (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.75	"	3.0889×10^{-3}
100	0.75	"	3.1905×10^{-3}
200	0.75	"	2.8923×10^{-3}
50	0.50	"	3.2173×10^{-3}
100	0.50	"	2.8033×10^{-3}
200	0.50	"	2.9817×10^{-3}
50	0.60	"	3.0971×10^{-3}
100	0.60	"	2.9575×10^{-3}
200	0.60	"	2.6219×10^{-3}

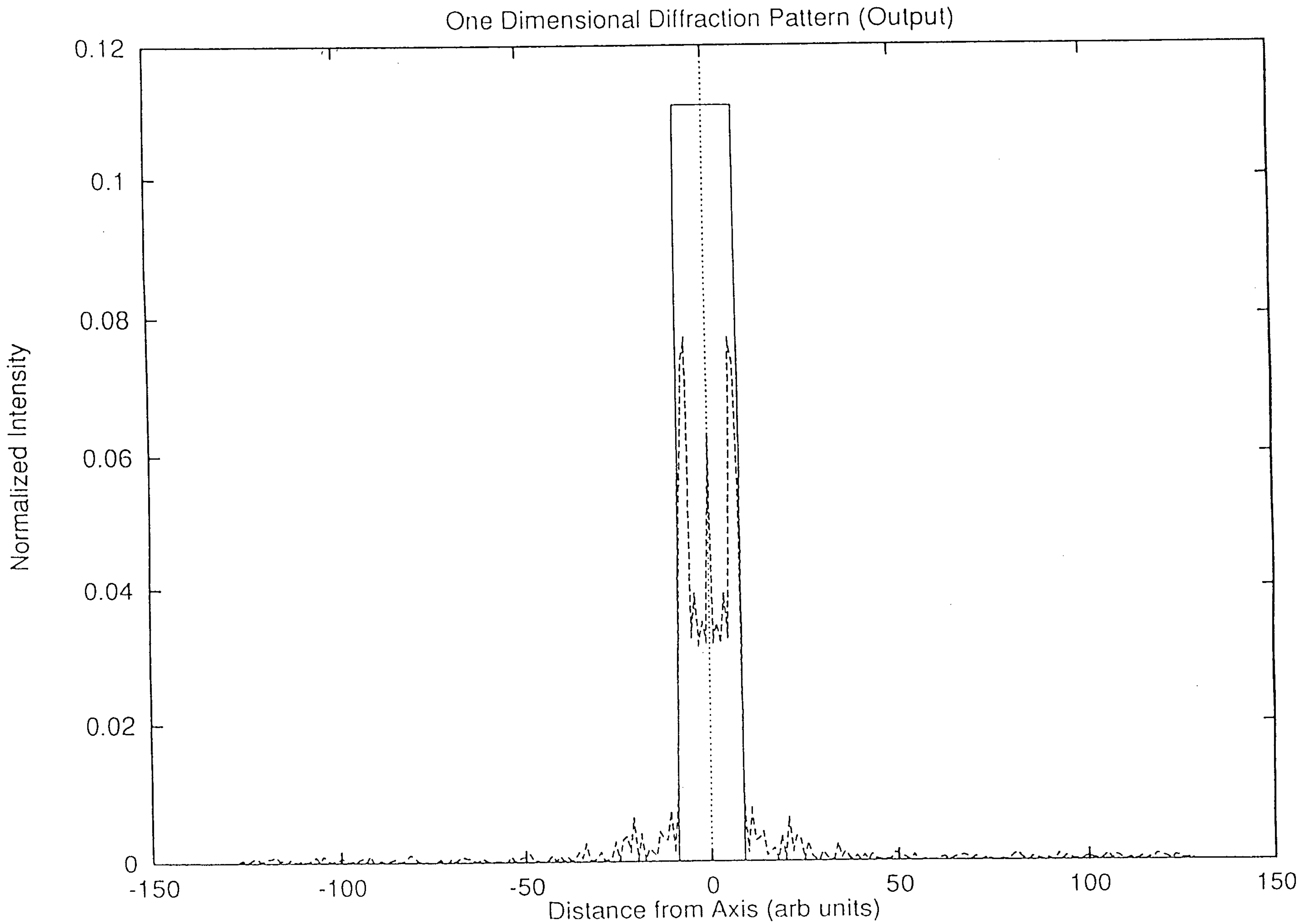


Fig. 4.34 9x9 One Dimensional Diffraction Pattern (Adaptive Mutation)

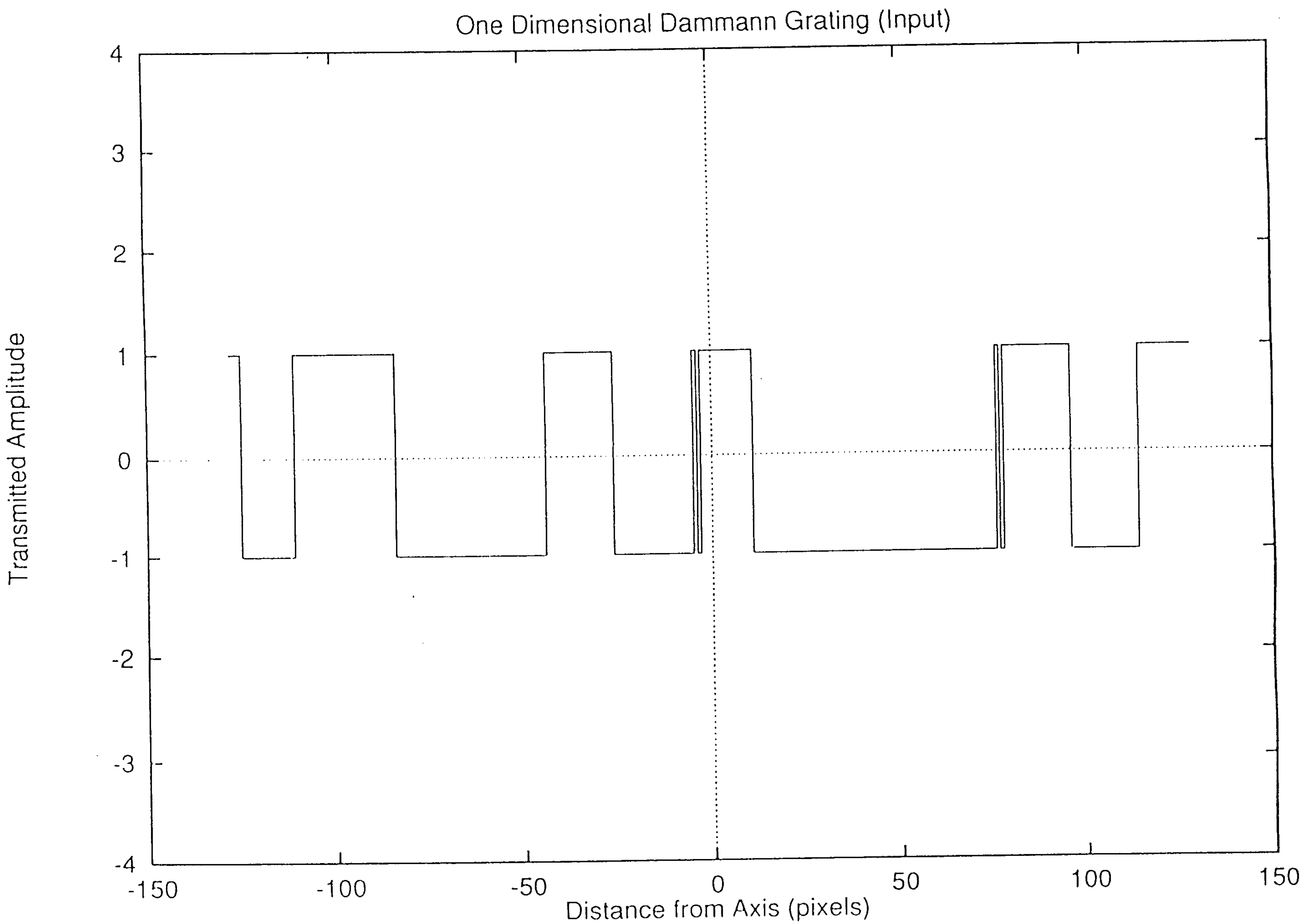


Fig. 4.35 9x9 One Dimensional Dammann Grating Structure (Adaptive Mutation)

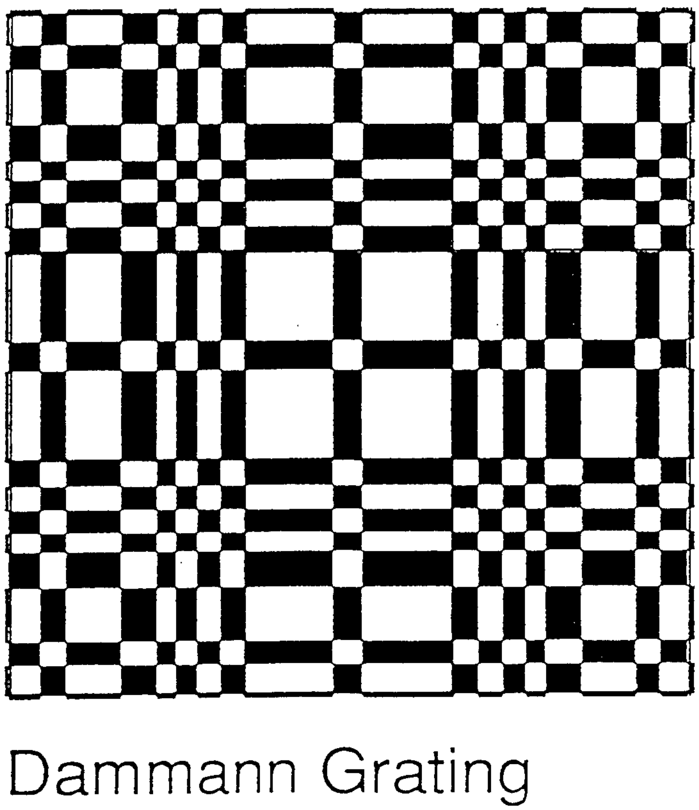


Fig. 4.36 9x9 Dammann Grating Cell (Adaptive Mutation)

11111111 11100000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00001011
11111111 11111111 00000000 00000000 00111111
11111111 11100000 00000000 01111111 11111111
11111111 11110000 00000000 00000000 00000000
00000000 00001111 11111111 11111100 00000000
00000000 00010111 3.1905e-03 1165 92140

Fig. 4.37 9x9 Objective Function Value (Adaptive Mutation)

Table 4.4 9x9 Dammann Grating Arrays with different Genetic Algorithms setting (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.75	0.001	2.9347×10^{-3}
100	0.75	0.001	3.1016×10^{-3}
200	0.75	0.001	3.1570×10^{-3}
50	0.50	0.10	4.3571×10^{-3}
100	0.50	0.10	4.7165×10^{-3}
200	0.50	0.10	5.2993×10^{-3}
50	0.60	0.001	2.9289×10^{-3}
100	0.60	0.001	3.1143×10^{-3}
200	0.60	0.001	3.1278×10^{-3}

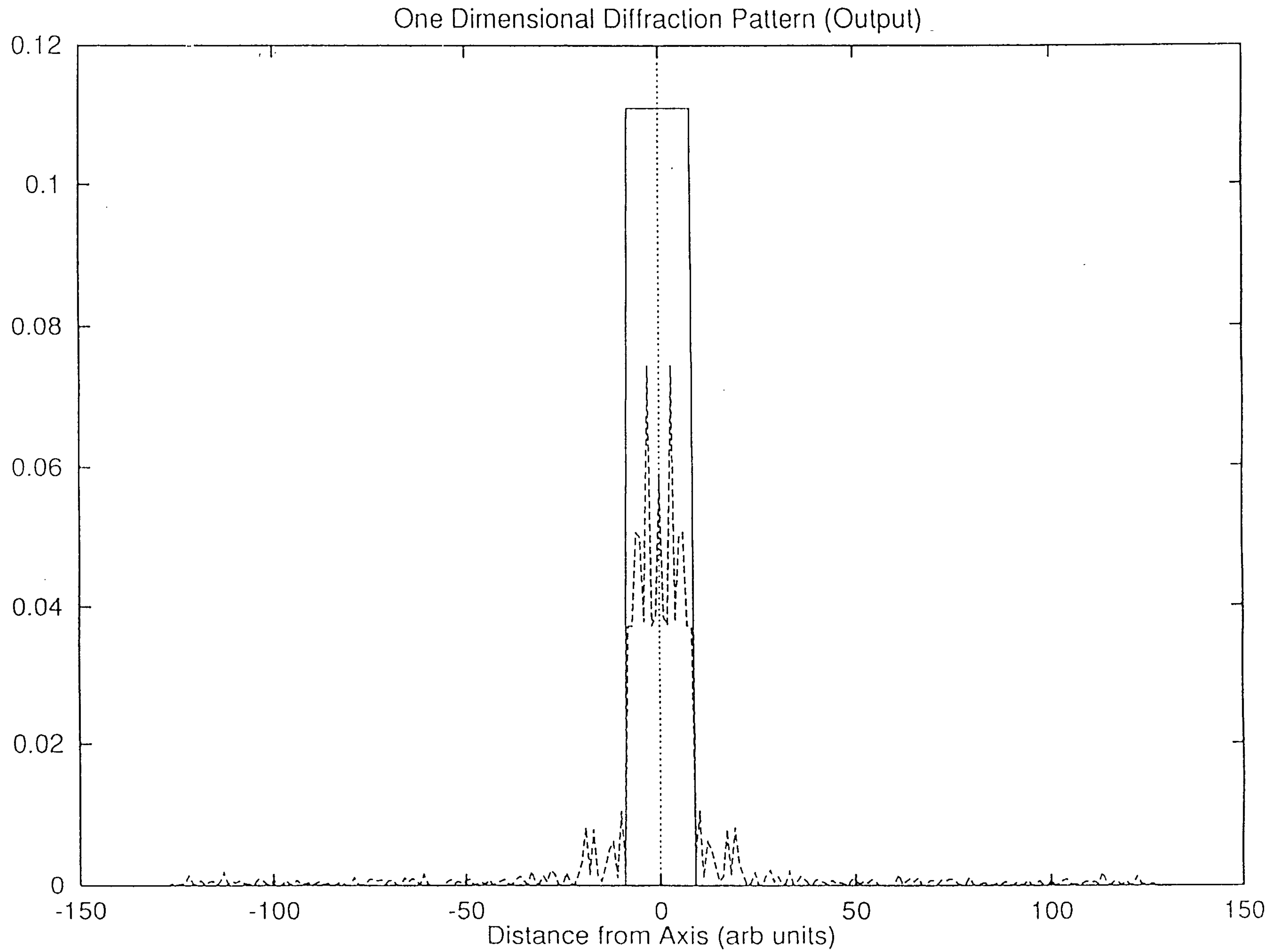


Fig. 4.38 9x9 One Dimensional Diffraction Pattern

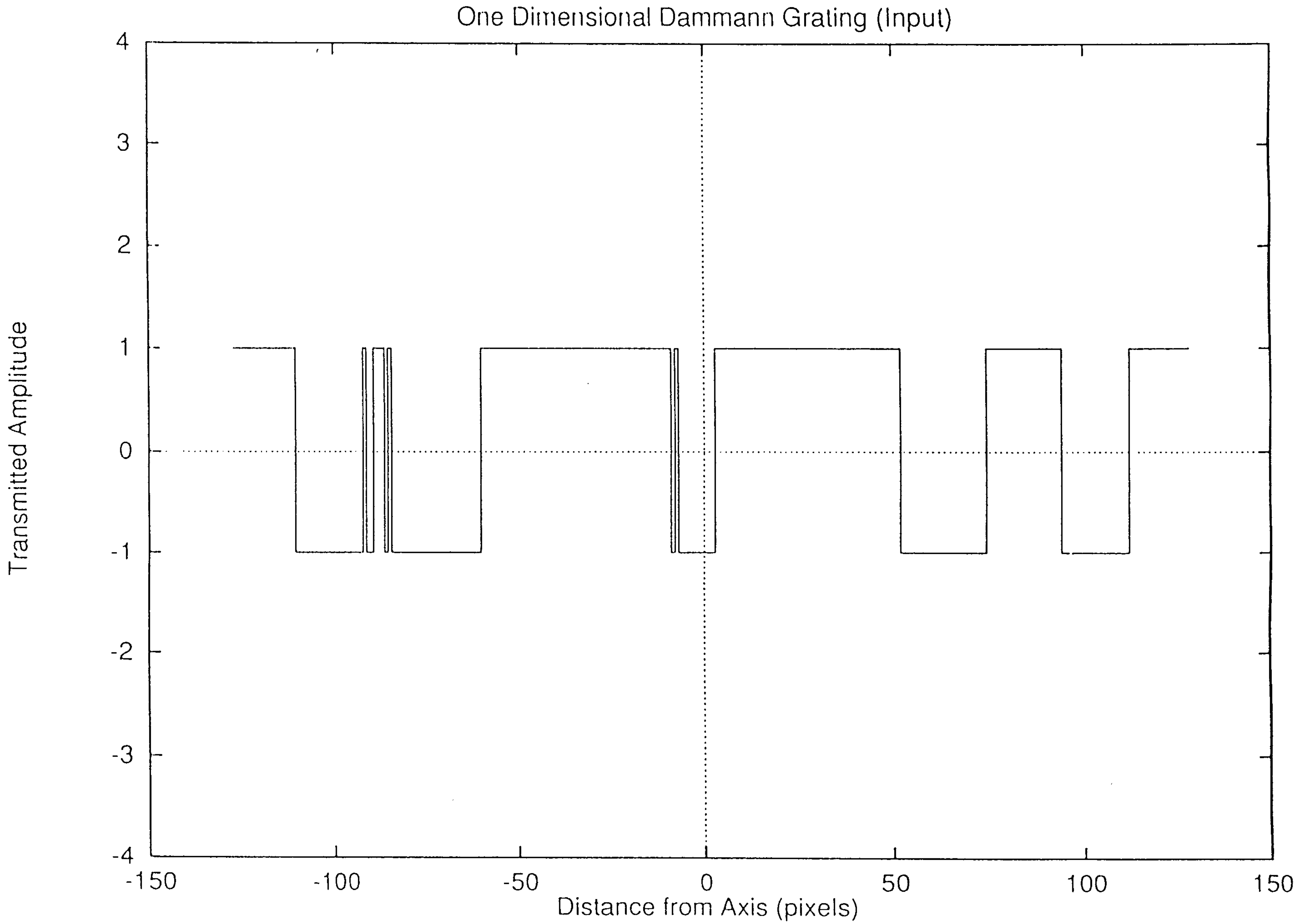


Fig. 4.39 9x9 One Dimensional Dammann Grating Structure

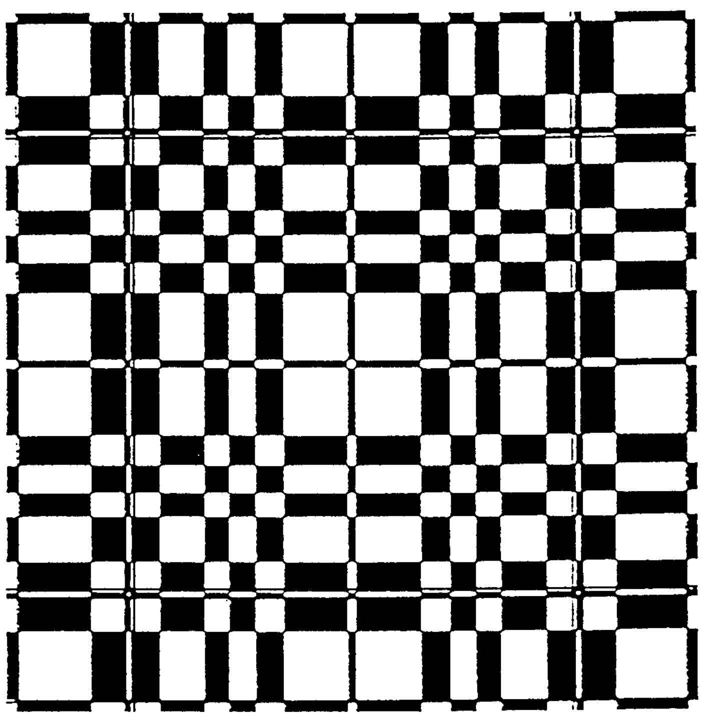


Fig. 4.40 9x9 Dammann Grating Cell

00011111 11111111 11111111 11111111 11111111
11111111 11110000 00000000 00000000 00111111
11111111 11111100 00000000 00000000 11111111
11111111 11111111 11111111 10000000 00000000
00001001 11010000 00000000 00000000 00001111
11111111 11111111 11111111 11111111 11111111
11111110 10000000 3.1016e-03 819 63228

Fig. 4.41 9x9 Objective Function Value

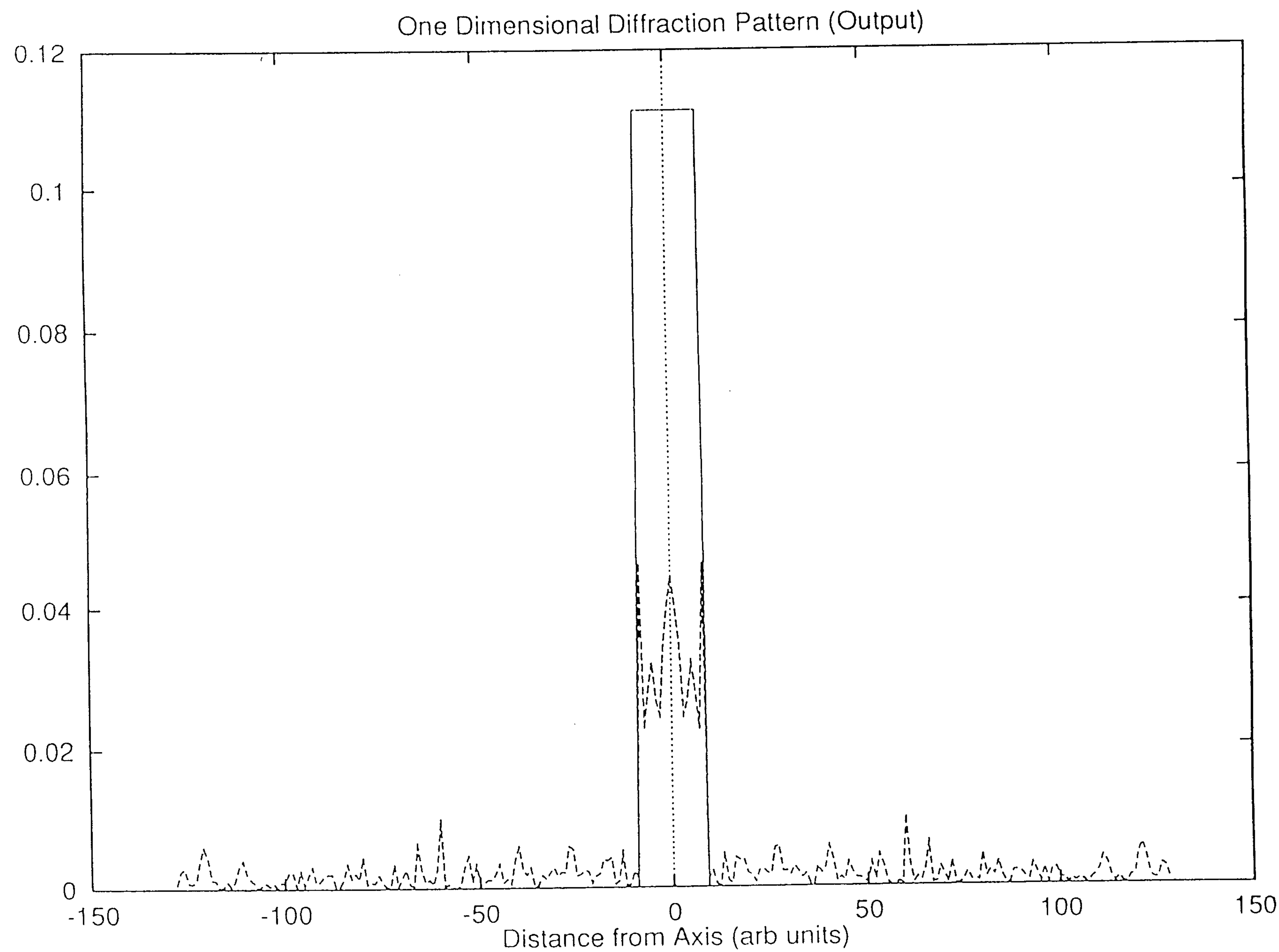


Fig. 4.42 9x9 One Dimensional Diffraction Pattern (High Mutation Rate)

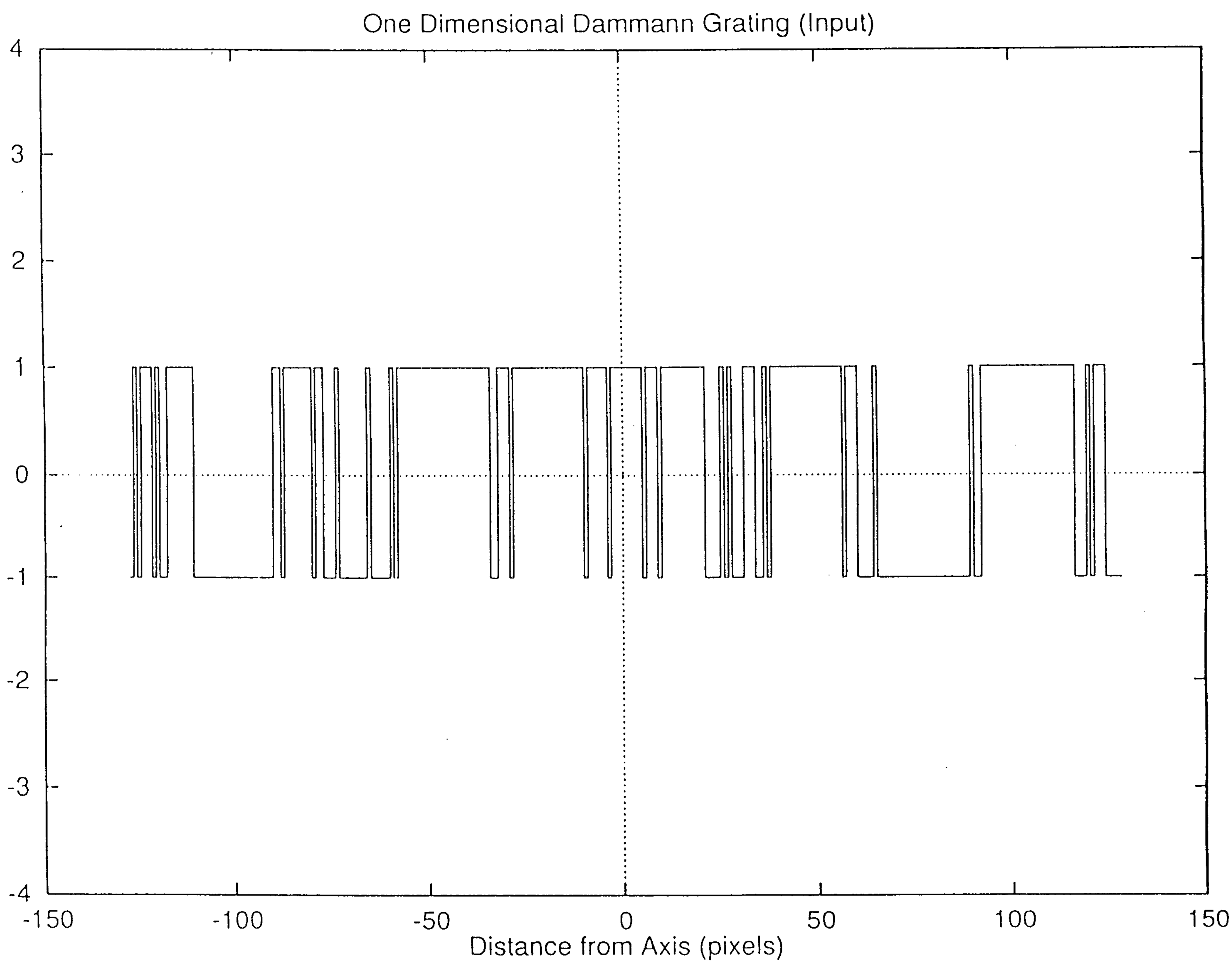


Fig. 4.43 9x9 One Dimensional Dammann Grating Structure (High Mutation Rate)

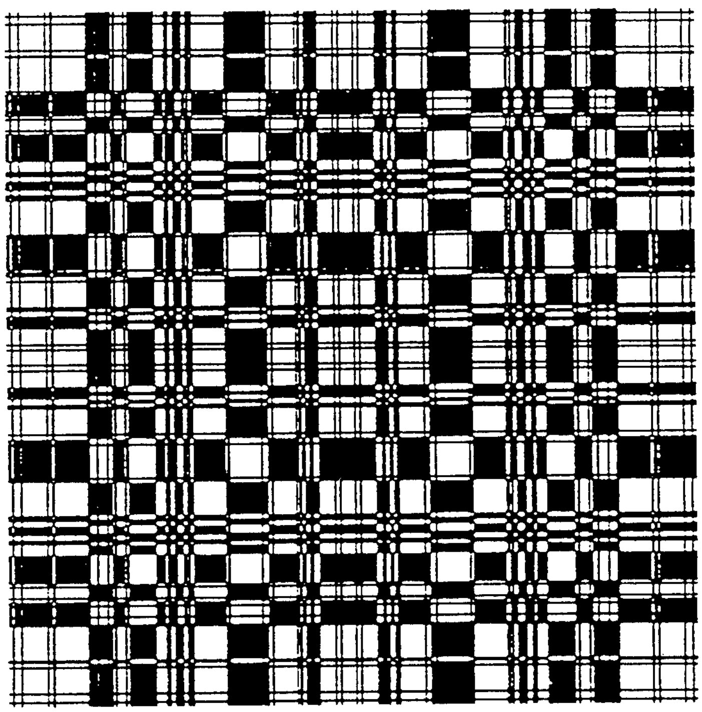


Fig. 4.44 9x9 Dammann Grating Cell (High Mutation Rate)

11111011 10111111 11111000 01010001 11001011
11111111 11111111 01110000 10000000 00000000
00000000 01001111 11111111 11111111 11110001
01110000 01011101 00111111 10000000 00000000
00000011 01111111 01100010 00000010 00001011
11111111 11111111 11111100 11101111 11111111
11111101 11110111 4.7165e-03 2979 297917

Fig. 4.45 9x9 Objective Function Value (High Mutation Rate)

Table 4.5 9x9 Dammann Grating Arrays with different Genetic Algorithms setting (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.75	"	2.9866×10^{-3}
100	0.75	"	3.0876×10^{-3}
200	0.75	"	3.2444×10^{-3}
50	0.50	"	2.9028×10^{-3}
100	0.50	"	2.9932×10^{-3}
200	0.50	"	3.1018×10^{-3}
50	0.60	"	3.4860×10^{-3}
100	0.60	"	3.0553×10^{-3}
200	0.60	"	2.8861×10^{-3}

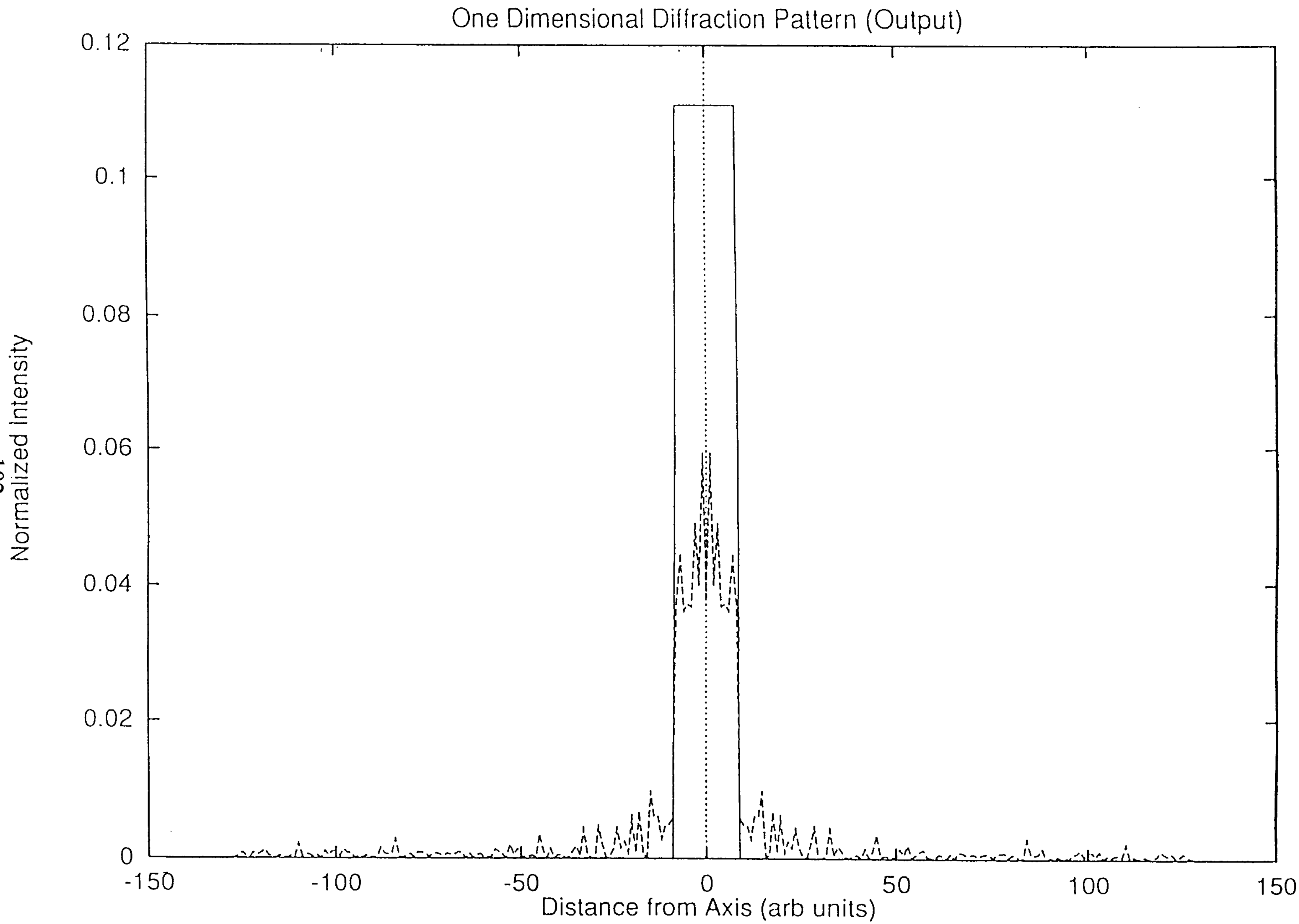


Fig. 4.46 9x9 One Dimensional Diffraction Pattern (Adaptive Mutation)

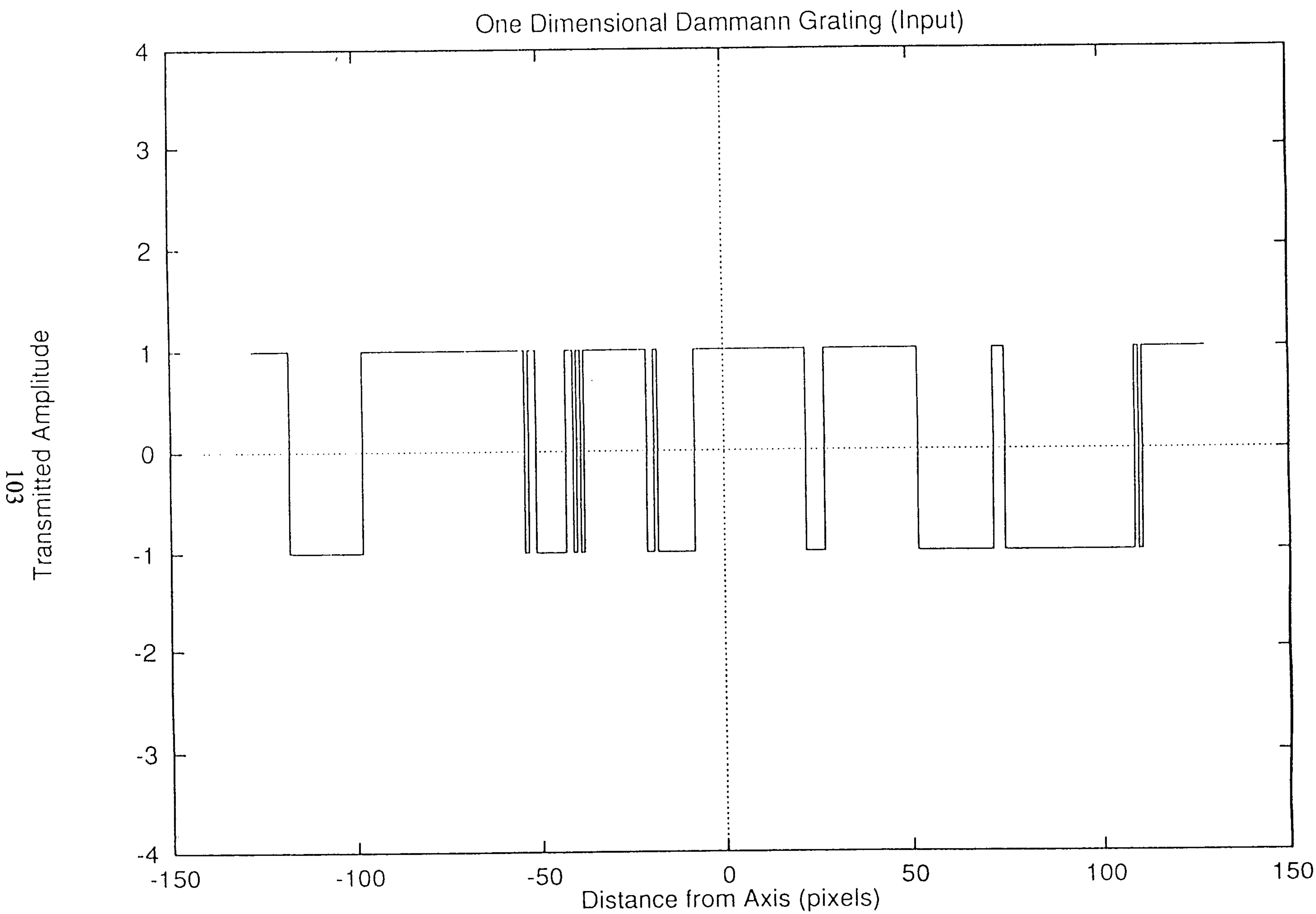


Fig. 4.47 9x9 One Dimensional Dammann Grating Structure (Adaptive Mutation)

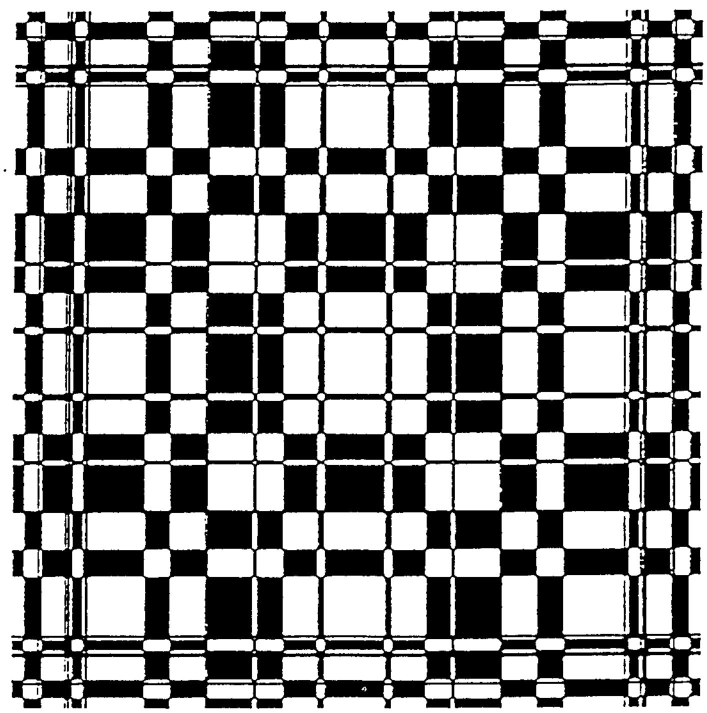


Fig. 4.48 9x9 Dammann Grating Cell (Adaptive Mutation)

11111111 11111111 11111100 00011111 11111111
11111111 11110000 00000000 00000000 11100000
00000000 00000000 00000000 00000101 11111111
11111111 11111111 11000000 00000000 00000011
11111111 11111111 11111111 11111111 11111111
11011000 00000110 10111111 11111111 11100100
00000000 11111111 3.4860e-03 9201 285867

Fig. 4.49 9x9 Objective Function Value (Adaptive Mutation)

Table 4.6 17x17 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.75	0.001	9.1284×10^{-4}
100	0.75	0.001	8.2262×10^{-4}
200	0.75	0.001	9.0223×10^{-4}
50	0.50	0.10	1.8442×10^{-3}
100	0.50	0.10	1.9068×10^{-3}
200	0.50	0.10	1.8448×10^{-3}
50	0.60	0.001	7.9977×10^{-4}
100	0.60	0.001	8.8623×10^{-4}
200	0.60	0.001	8.7483×10^{-4}

diffraction pattern as shown in figure 4.38 is highly resolved. Shown in figures 4.42, 4.43, 4.44, and 4.45 are results of the simulation with high mutation rate. In the case of the one point crossover and adaptive mutation, the results obtained are shown in table 4.5 and in figures 4.46, 4.47, 4.48, and 4.49 respectively. Again, it was observed that siimilar results were obtained when lower standard mutation rate and and adaptive mutation were used. Two or one point crossover scheme made no significant difference in the results that were obtained.

4.2 17 x 17 Amplitude Dammann Gratings

This amplitude Dammann Grating element was also generated using the modified GenesYs 1.0 package. Here, two point crossover scheme, one point crossover scheme, standard and adaptive mutation mechanisms were also employed. For a 17x17 size Dammann Gratings, 512 bit strings (0,1) were required, and these strings were randomly generated. The simulation of the element using different population sizes were done at 300000 number of trials. In terms of number of generations, a 100 size population is equivalent to 3000 generations. In comparision to a 9x9 size Dammann Gratings, the degree of computation difficulties and time increases with this element. The resolution of the diffraction patterns decreases. The different settings used here, such as, population size, crossover rate, and mutation rate are shown in table 4.6 for the two-point standard mutation case. The results obtained with population size of 100, 75 per cent crossover rate, and standard mutation of 0.1 per cent are shown in figures 4.50, 4.51, 4.52, 4.53 and 4.54 respectively. Shown in figures

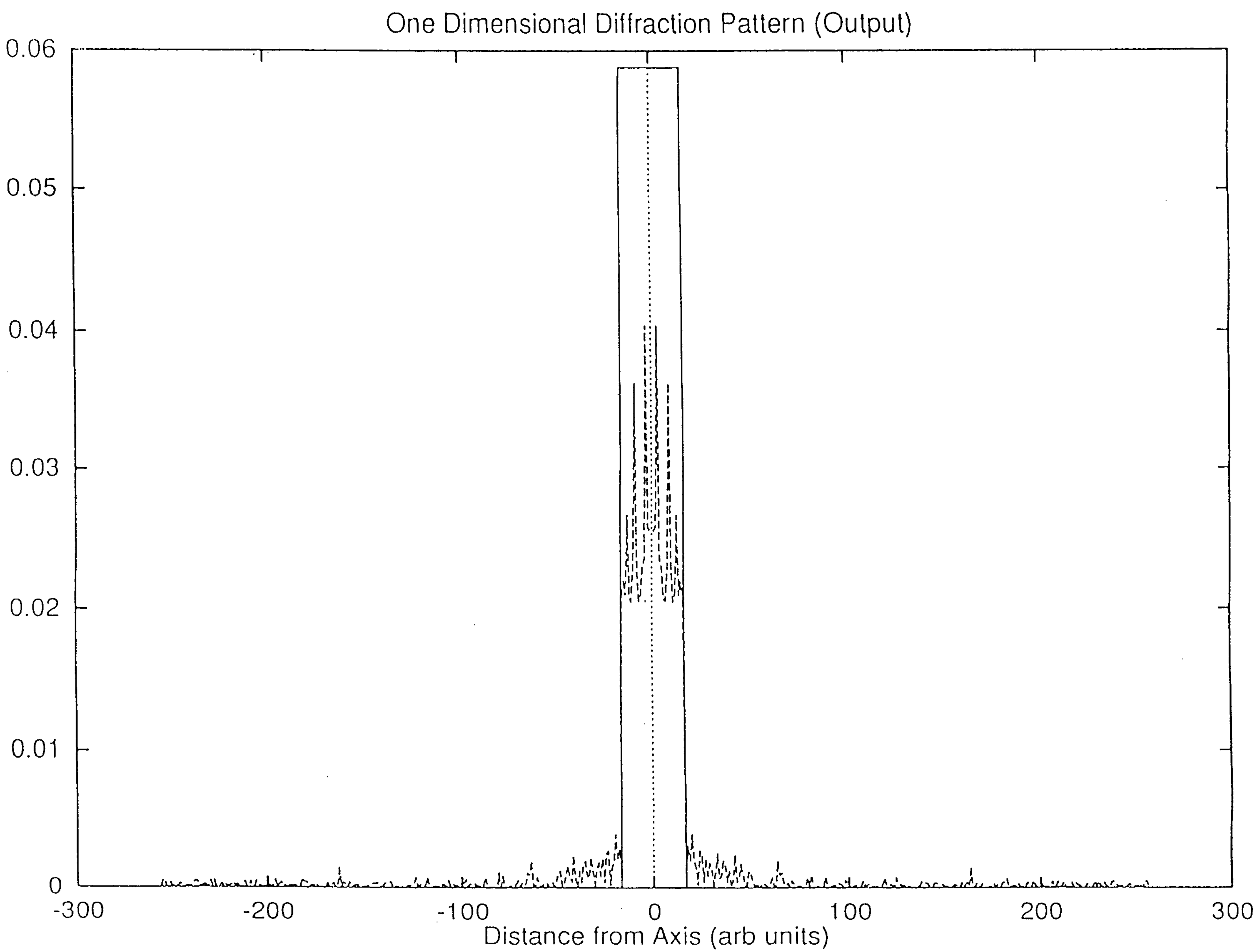


Fig. 4.50 17x17 One Dimensional Diffraction Pattern

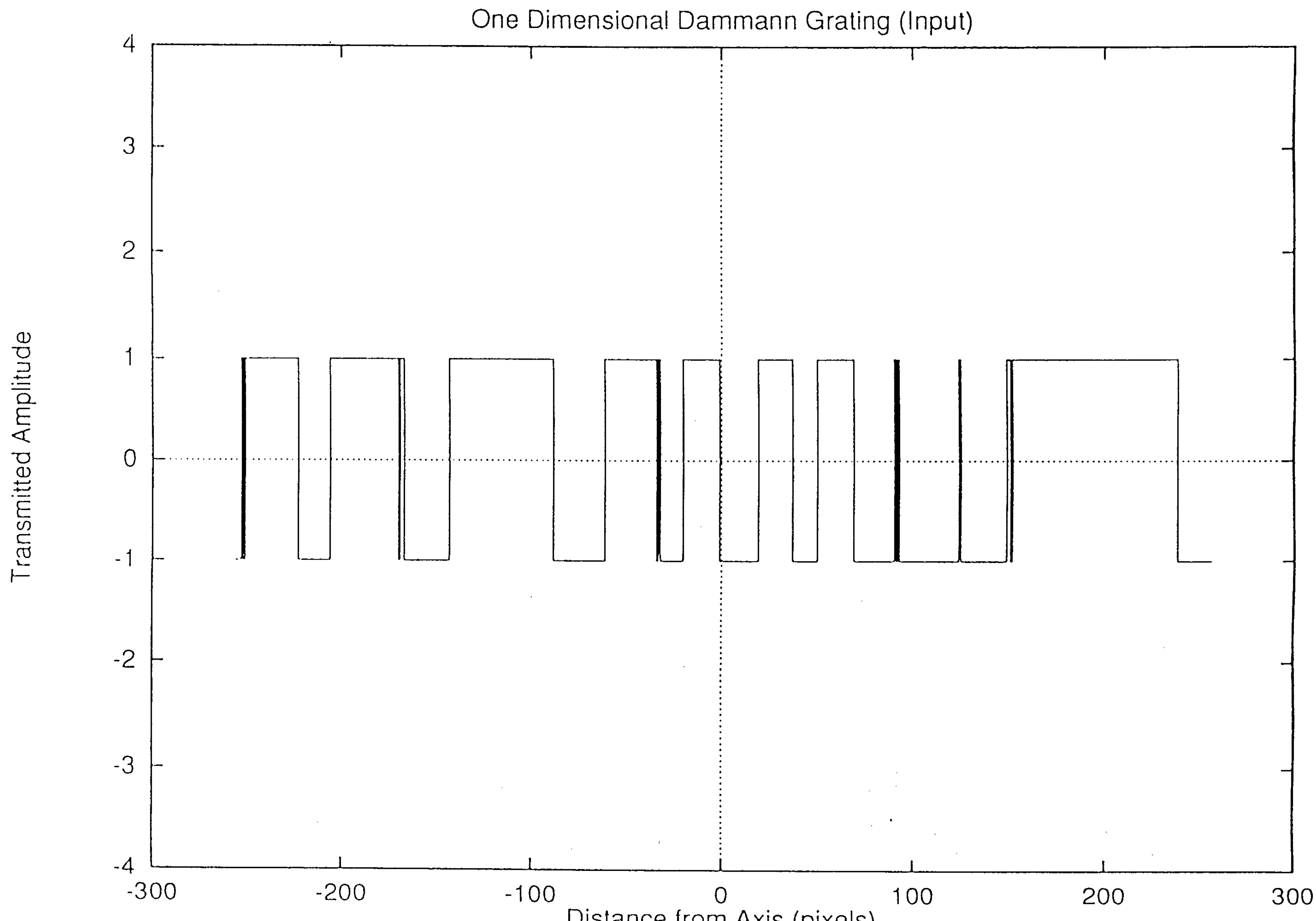


Fig. 4.51 17x17 One Dimensional Dammann Grating Structure

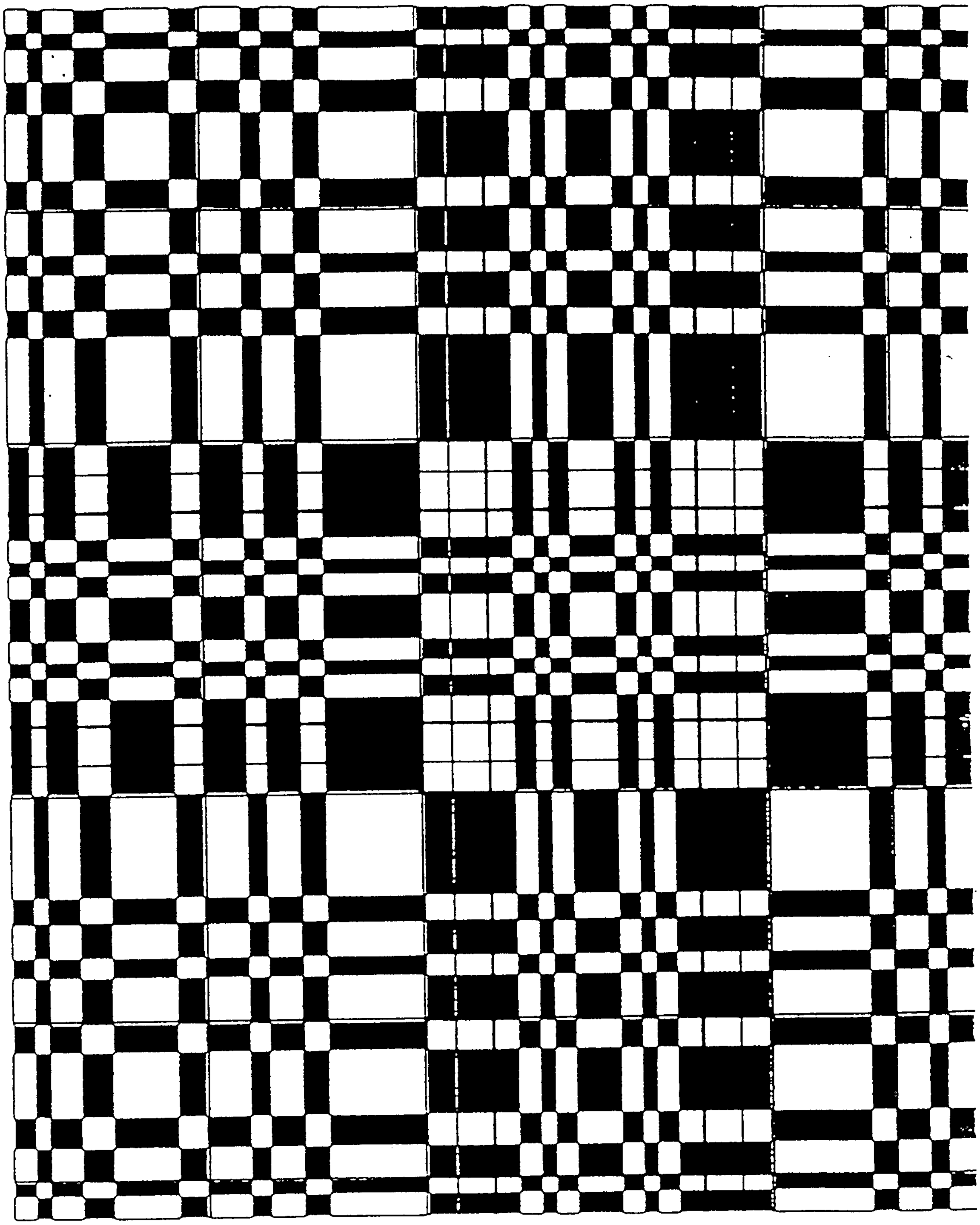


Fig. 4.52 17x17 Dammann Grating Cell

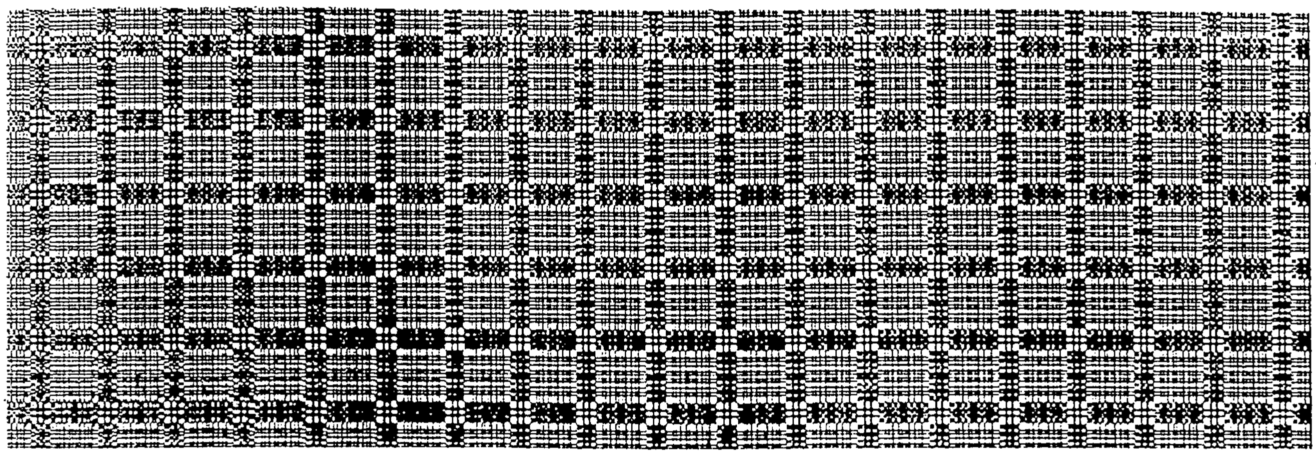


Fig. 4.53 17x17 Fabricated Amplitude Dammann Grating Device

11111111 11111111 11111111 11111111 11111100
00000000 00000000 00000101 10011000 00000000
00000000 00000000 00000000 00000000 00000000
00001111 11111111 11111011 00000000 00000000
00000000 00000000 00000000 11111111 11111111
01000000 00000000 00000001 11111111 11111000
00000000 00000000 00011111 11111111 11110000
00000000 00000000 00000000 00000000 00000000
00000000 00000111 11111111 11111111 11111111
00000000 00000000 00000000 00000000 00000011
11111111 11111111 11111100 00000000 00000001
00111111 11111111 11110000 00000000 00000000
00000110 01111111 11111111 11111111 8.5105e-
04 3542 282870

Fig. 4.54 17x17 Objective Function Value

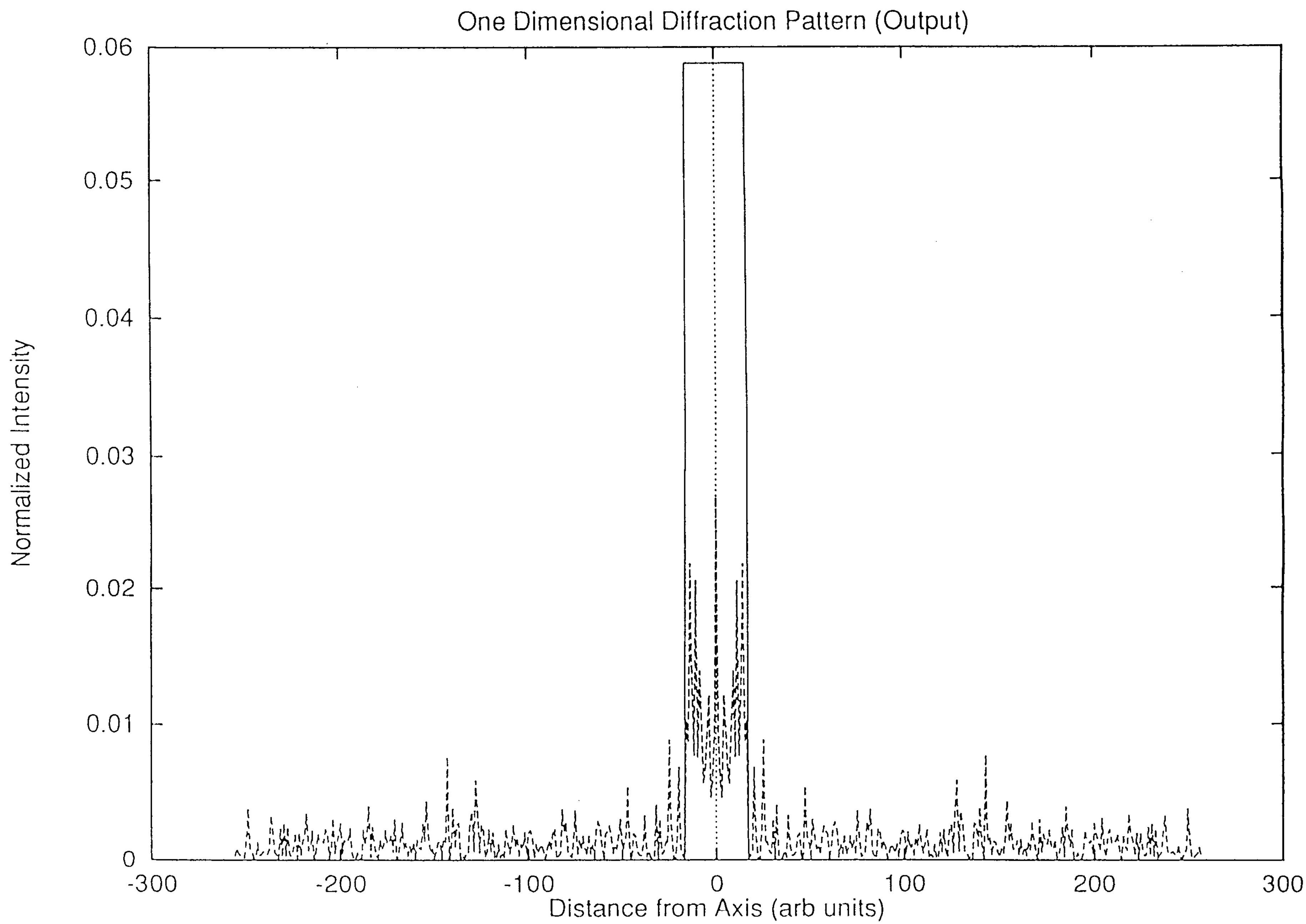


Fig. 4.55 17x17 One Dimensional Diffraction Pattern (High Mutation Rate)

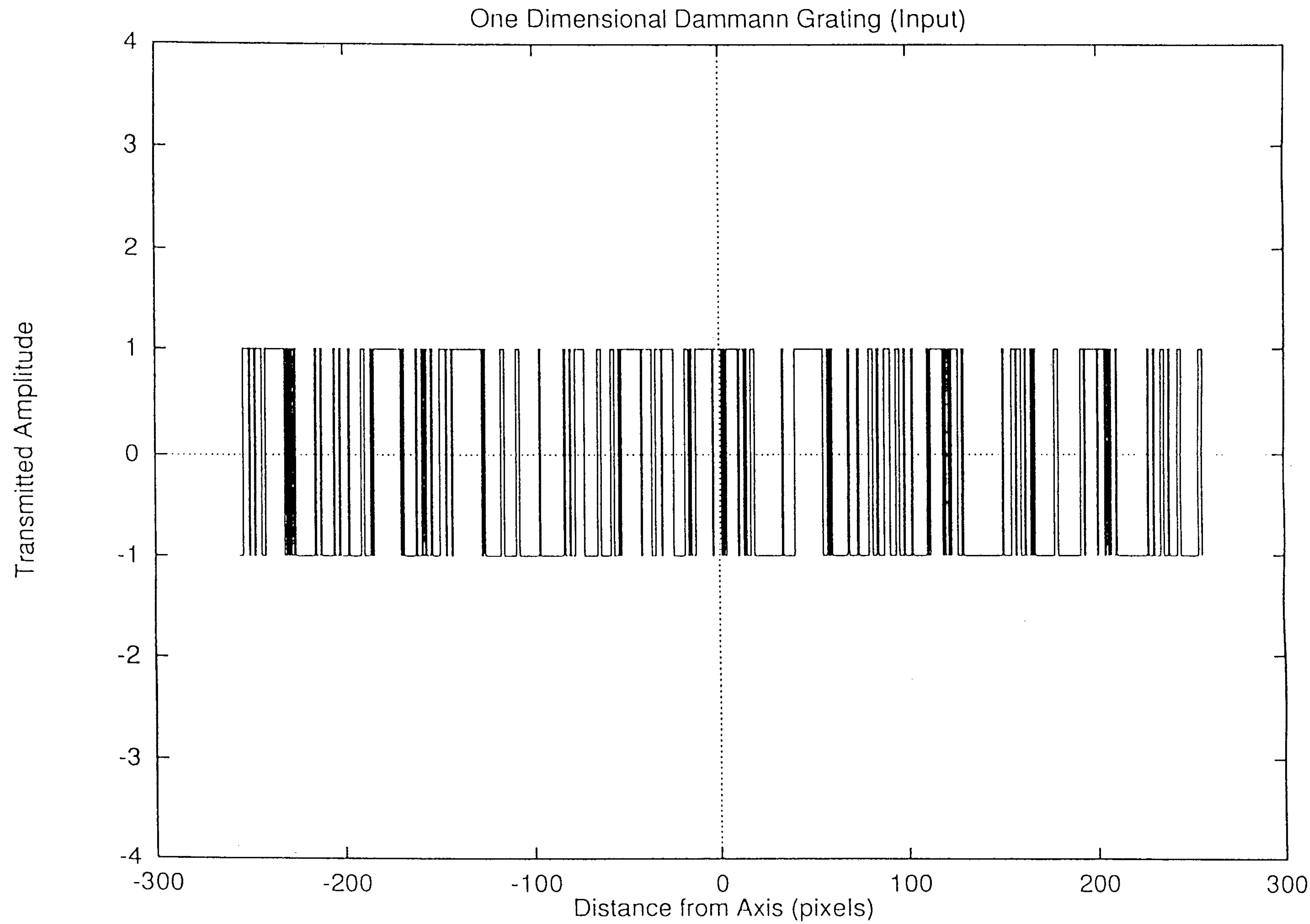


Fig.4.56 17x17 One Dimensional Dammann Grating Structure (High Mutation Rate)

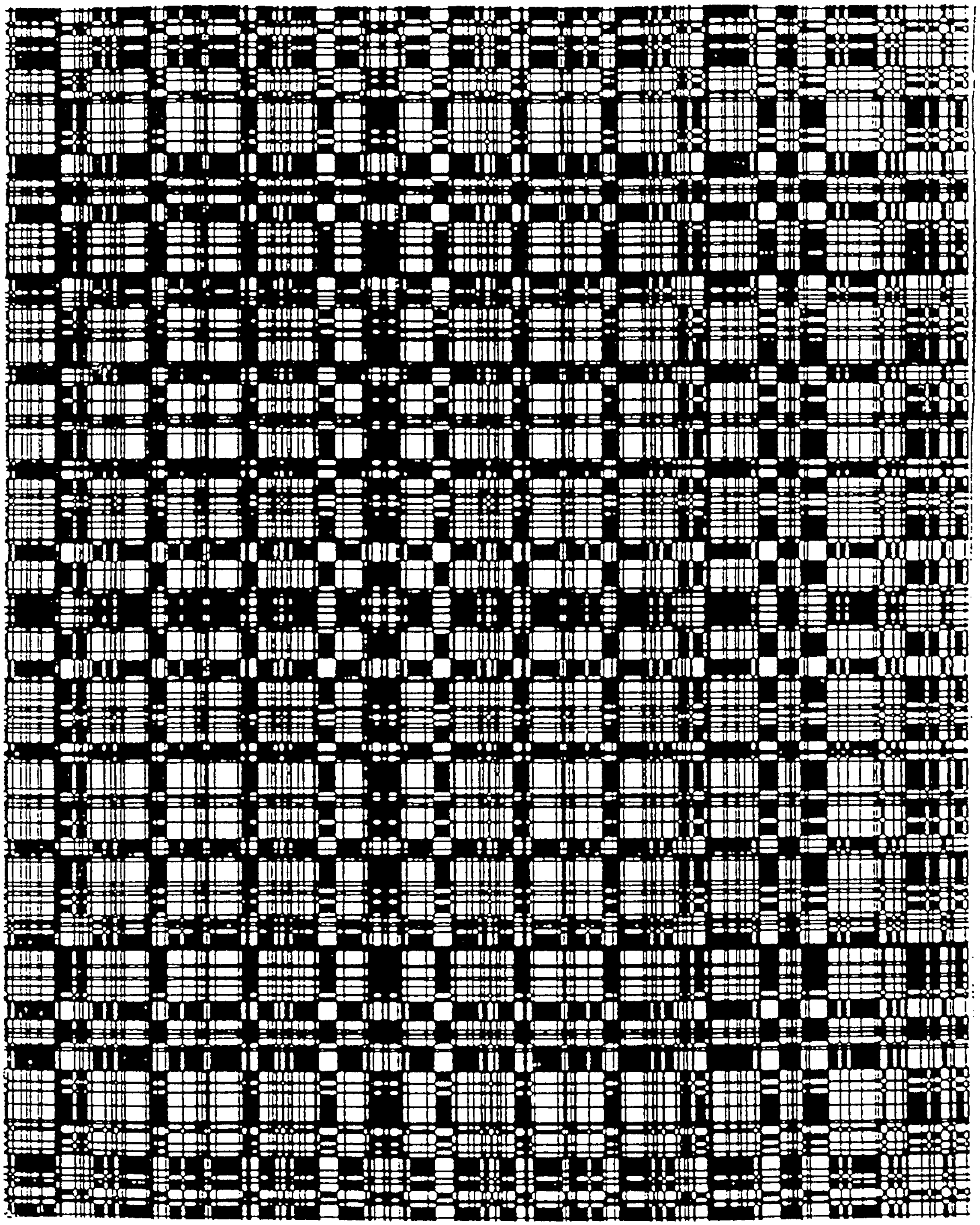


Fig..4.57 17x17 High Mutation Rate Dammann Grating Cell

10101111 11011010 01100000 00000000 00100000
01111111 11111111 00101000 00000100 00100000
11001000 11100011 00100010 00000010 11111101
01011100 10000000 00000000 00000100 00110110
01001010 00000000 01100000 00000001 10111111
01110101 00100000 00000000 00010010 00110010
00011000 00000011 00111011 01110011 11111111
01010100 00000000 10010000 00100100 00100000
01100010 11111111 11111101 00000010 01010010
00011101 10111111 11111111 10100000 00011000
00011000 00000001 00000000 00001001 00111110
00000011 00000110 01011111 11111011 11100111
01111110 00000110 10011111 11110111 1.8375e-03
2537 253724

Fig. 4.58 17x17 Objective Function Value (High Mutation Rate)

Table 4.7 17x17 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0) (Two Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.75	"	8.4271×10^{-4}
100	0.75	"	9.7167×10^{-4}
200	0.75	"	8.9792×10^{-4}
50	0.50	"	8.4991×10^{-4}
100	0.50	"	8.0058×10^{-4}
200	0.50	"	8.7639×10^{-4}
50	0.60	"	8.8887×10^{-4}
100	0.60	"	8.4602×10^{-4}
200	0.60	"	8.3629×10^{-4}

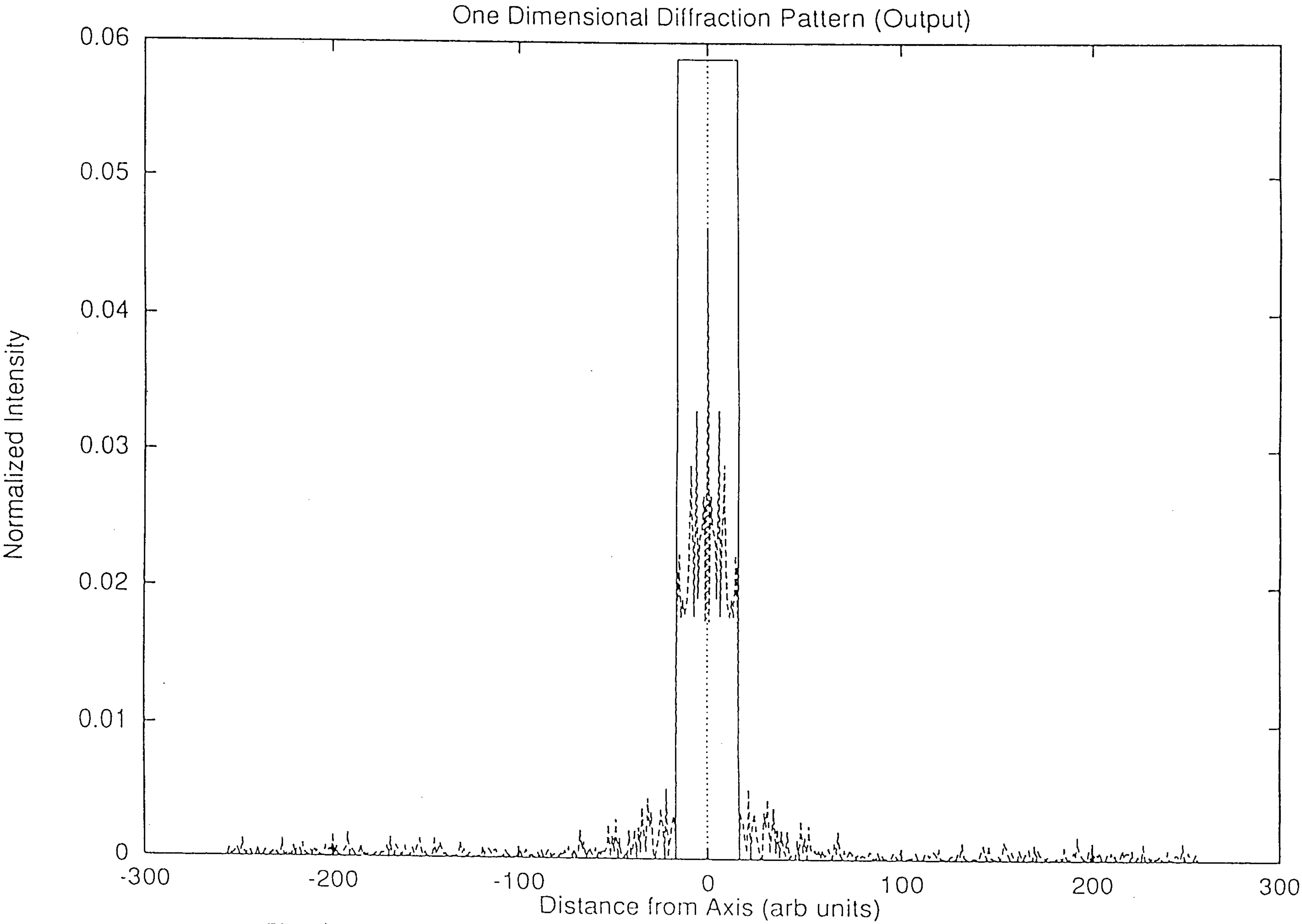


Fig. 4.59 17x17 Adaptive Mutation One Dimensional Diffraction Pattern

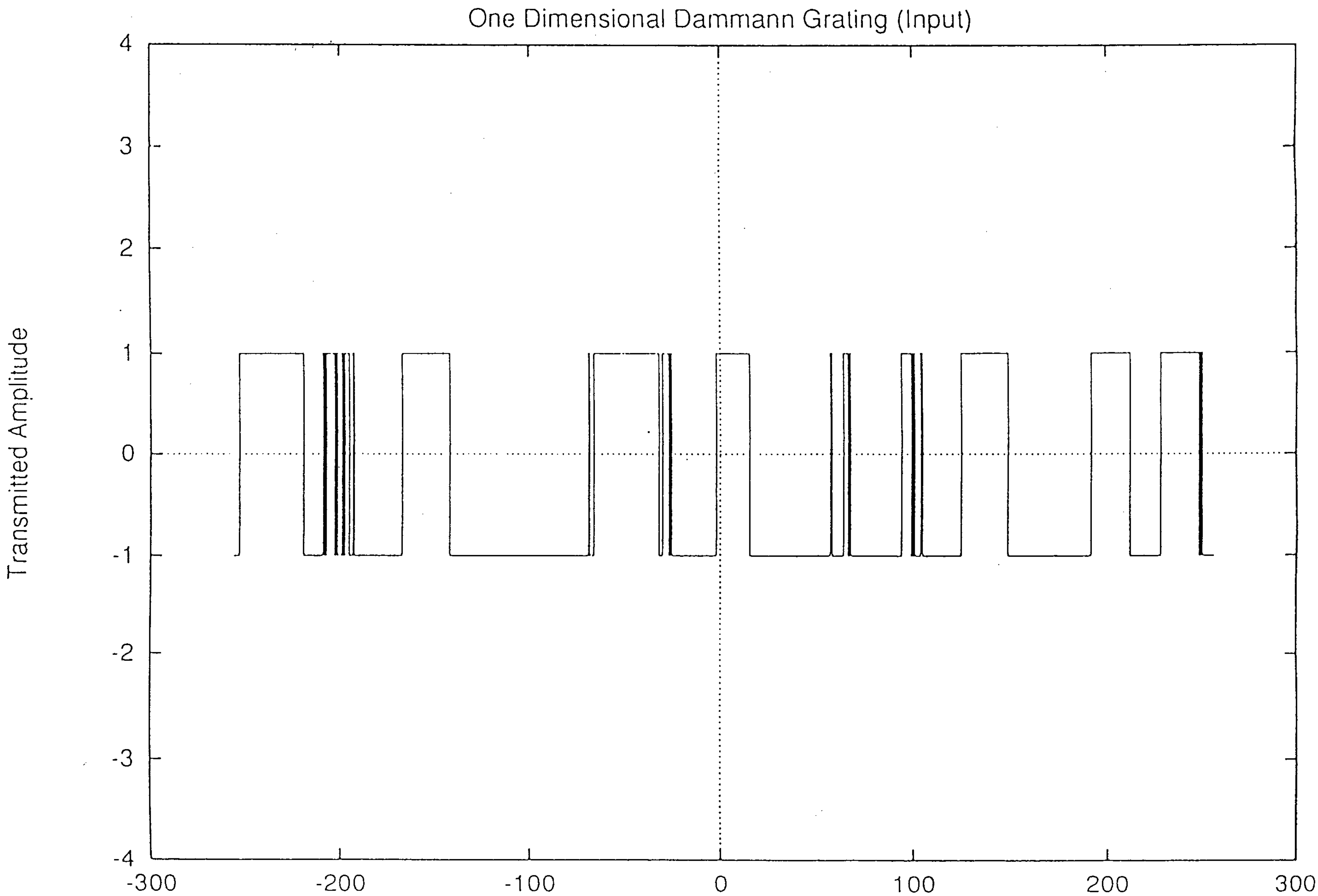


Fig. 4.60 17x17 Adaptive Mutation One Dimensional Dammann Grating

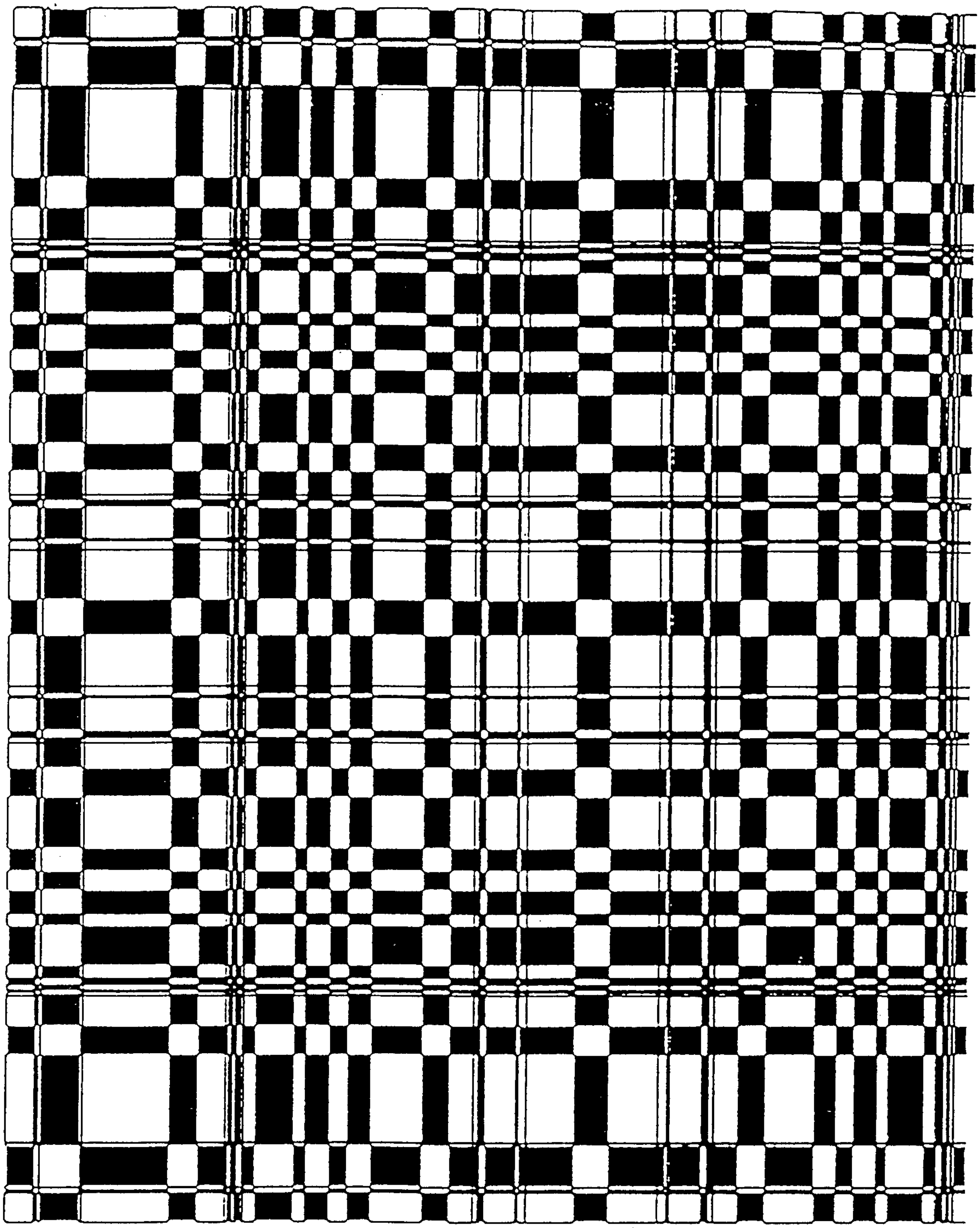


Fig. 4.61 17x17 Adaptive Mutation Dammann Grating Cell

11111111 11111110 00000000 00000000 00000000
00000000 00000000 01000000 11010000 00000000
00000000 00000011 11101000 10000000 00000000
00000111 11111111 11111111 11111000 00000000
00000000 00000000 00000000 00000000 11111111
11111111 11110000 00000000 00001111 11111111
11111111 01000000 00011111 11111111 11111111
11111111 11111000 00000001 01111010 01011001
00000000 00000000 00000000 01111111 11111111
11111111 11000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000
00010011 11111111 11111111 11111111 11111111
00111010 00000000 00000000 00000011 9.7167e-
04 825 64376

Fig. 4.62 17x17 Adaptive Mutation Objective Function Value

Table 4.8 17x17 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.75	0.001	$8.2270x10^{-4}$
100	0.75	0.001	$8.9474x10^{-4}$
200	0.75	0.001	9.479510^{-4}
50	0.50	0.10	$1.7493x10^{-3}$
100	0.50	0.10	$1.8217x10^{-3}$
200	0.50	0.10	$1.8670x10^{3}$
50	0.60	0.001	$9.3519x10^{-4}$
100	0.60	0.001	$8.3775x10^{-4}$
200	0.60	0.001	$8.8555x10^{-4}$

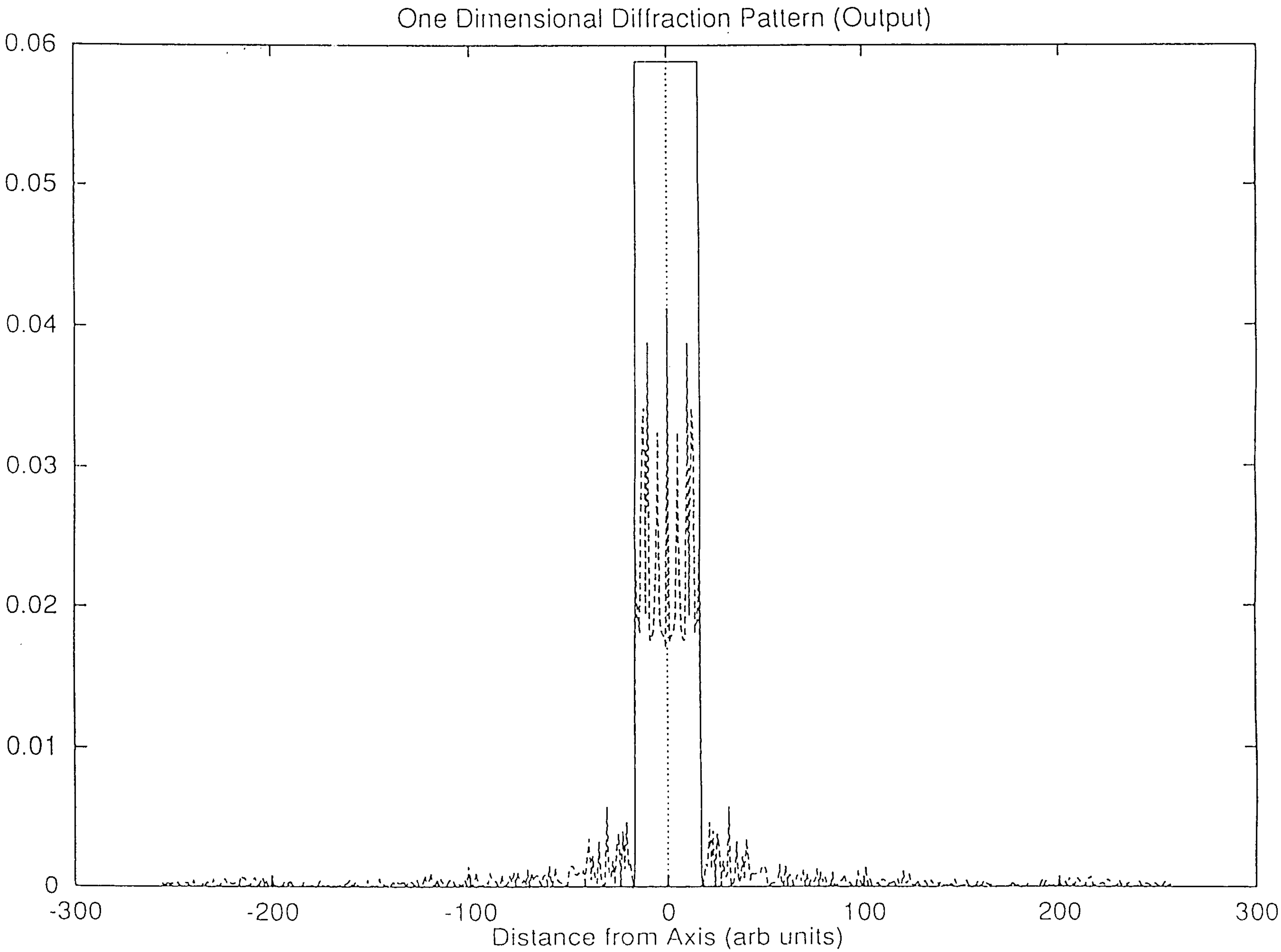


Fig. 4.63 17x17 One Dimensional Diffraction Pattern

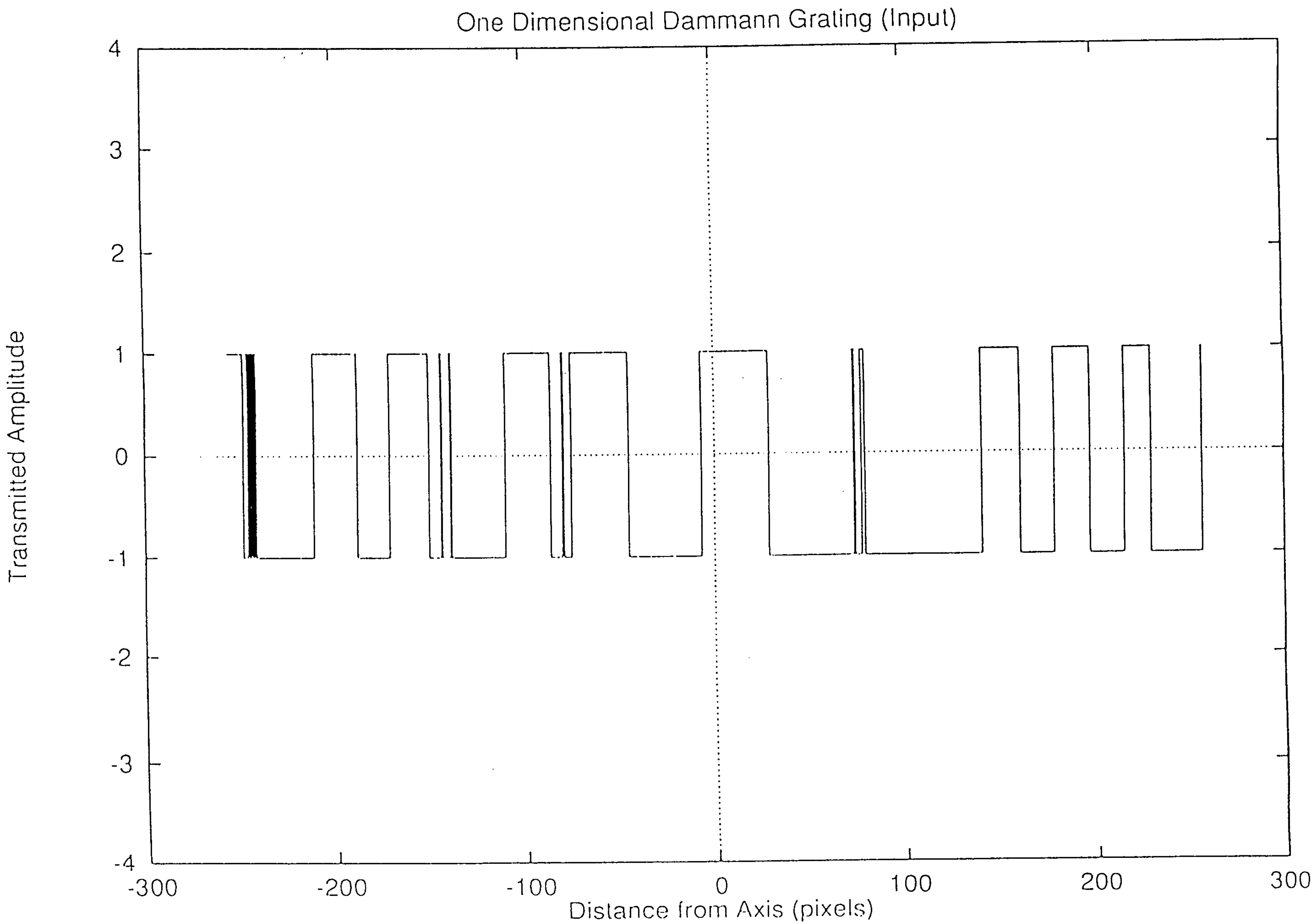


Fig. 4.64 17x17 One Dimensional Dammann Grating

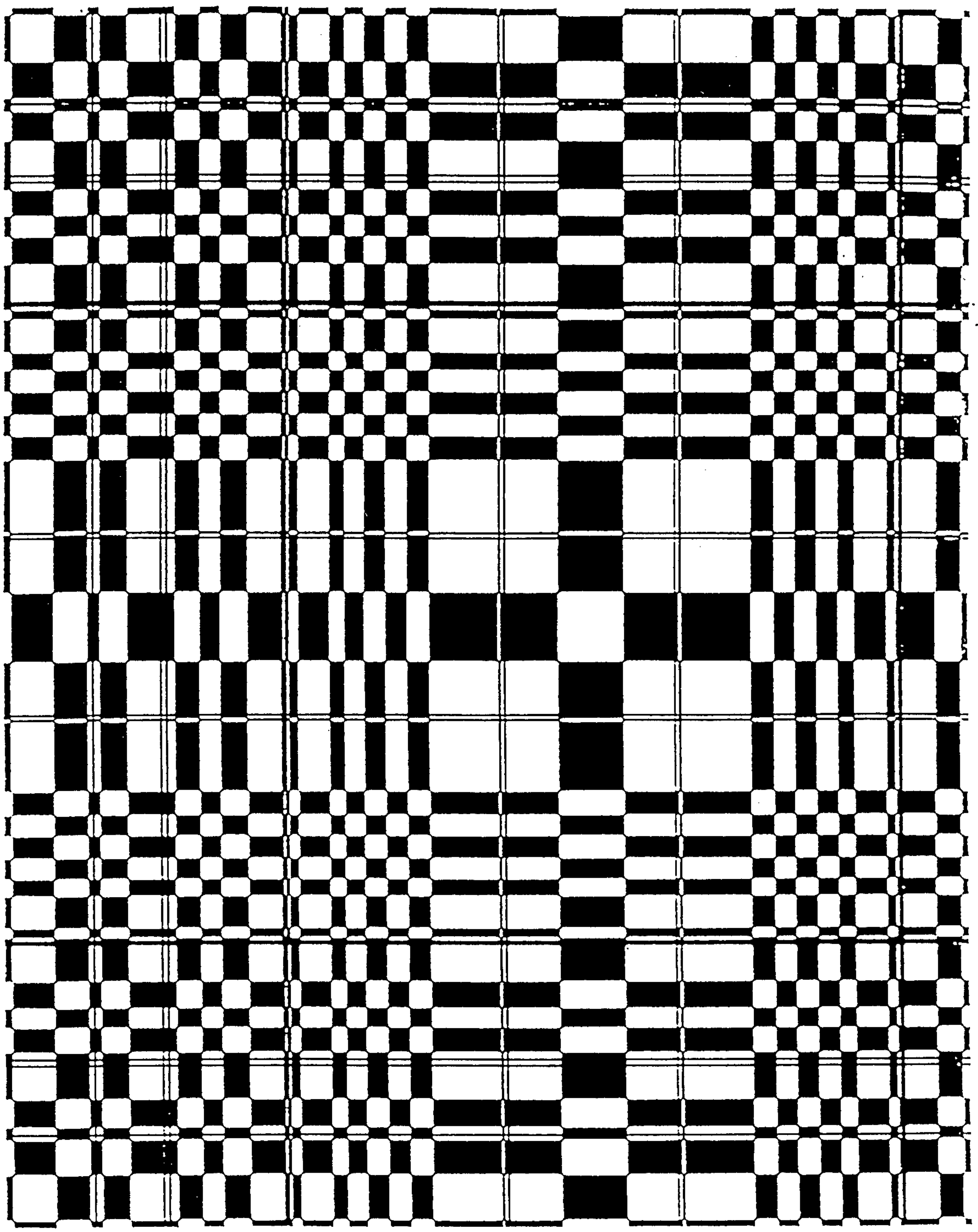


Fig. 4.65 17x17 Dammann Grating Cell

11111111 11111111 11111111 11111000 00000000
00000000 00000000 00000000 00000000 00100011
00000000 00000000 00000000 00000000 00000000
00000000 00000000 00001111 11111111 11111111
00000000 00000000 00111111 11111111 11111000
00000000 00000001 11111111 11111000 00000000
00000000 00000000 11111111 00101010 00000000
00000000 00000000 00000111 11111111 11111111
11110000 00000000 00000111 11111111 11111111
11000000 10000100 00000000 00000000 00000000
00111111 11111111 11111111 11000000 10000111
11111111 11111111 11111111 11100000 00000000
00000000 00000000 00000000 01111111 9.4795e-
04 1663 268311

Fig. 4.66 17x17 Objective Function Value

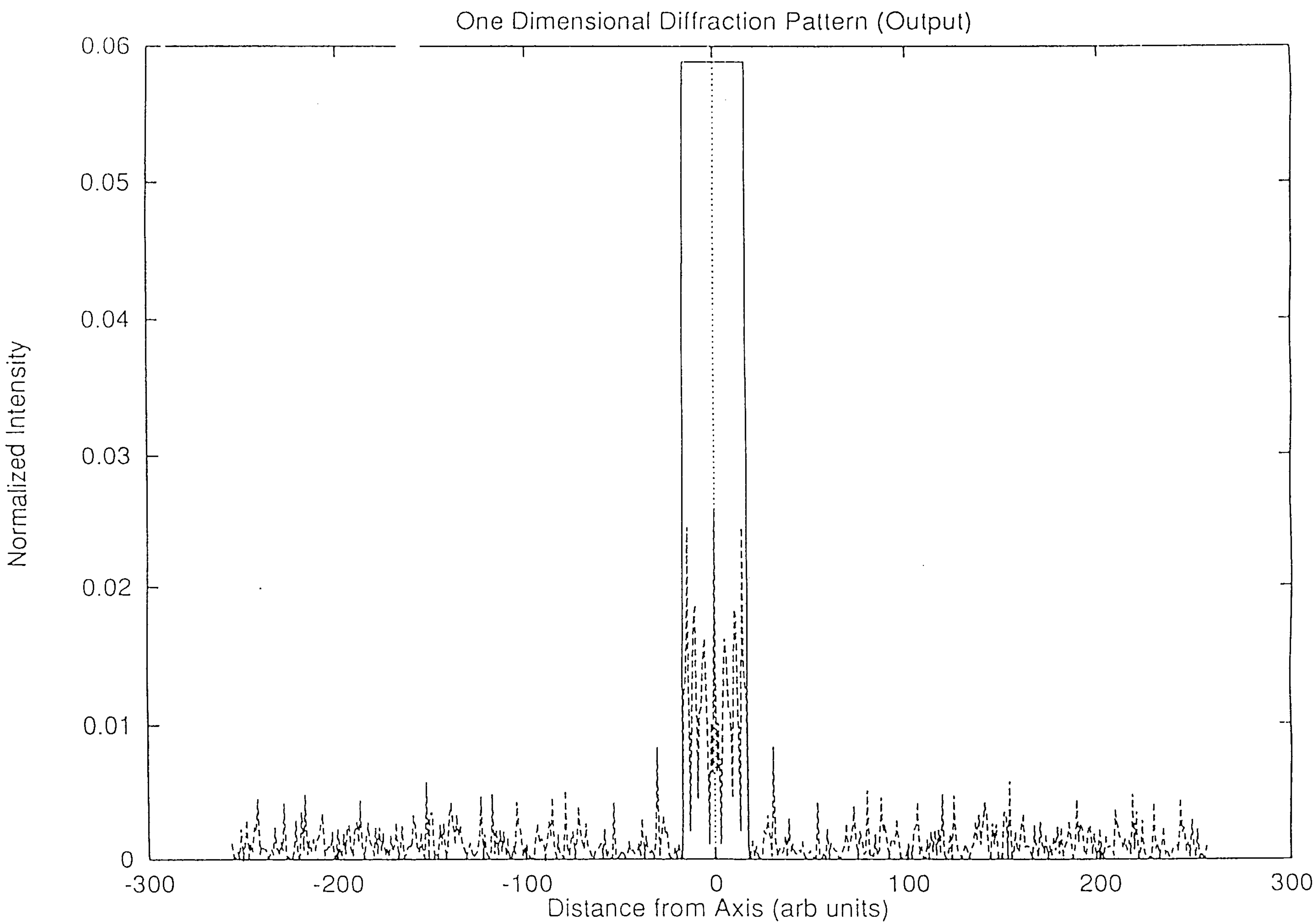


Fig. 4.67 17x17 High Mutation Rate One Dimensional Diffraction Pattern

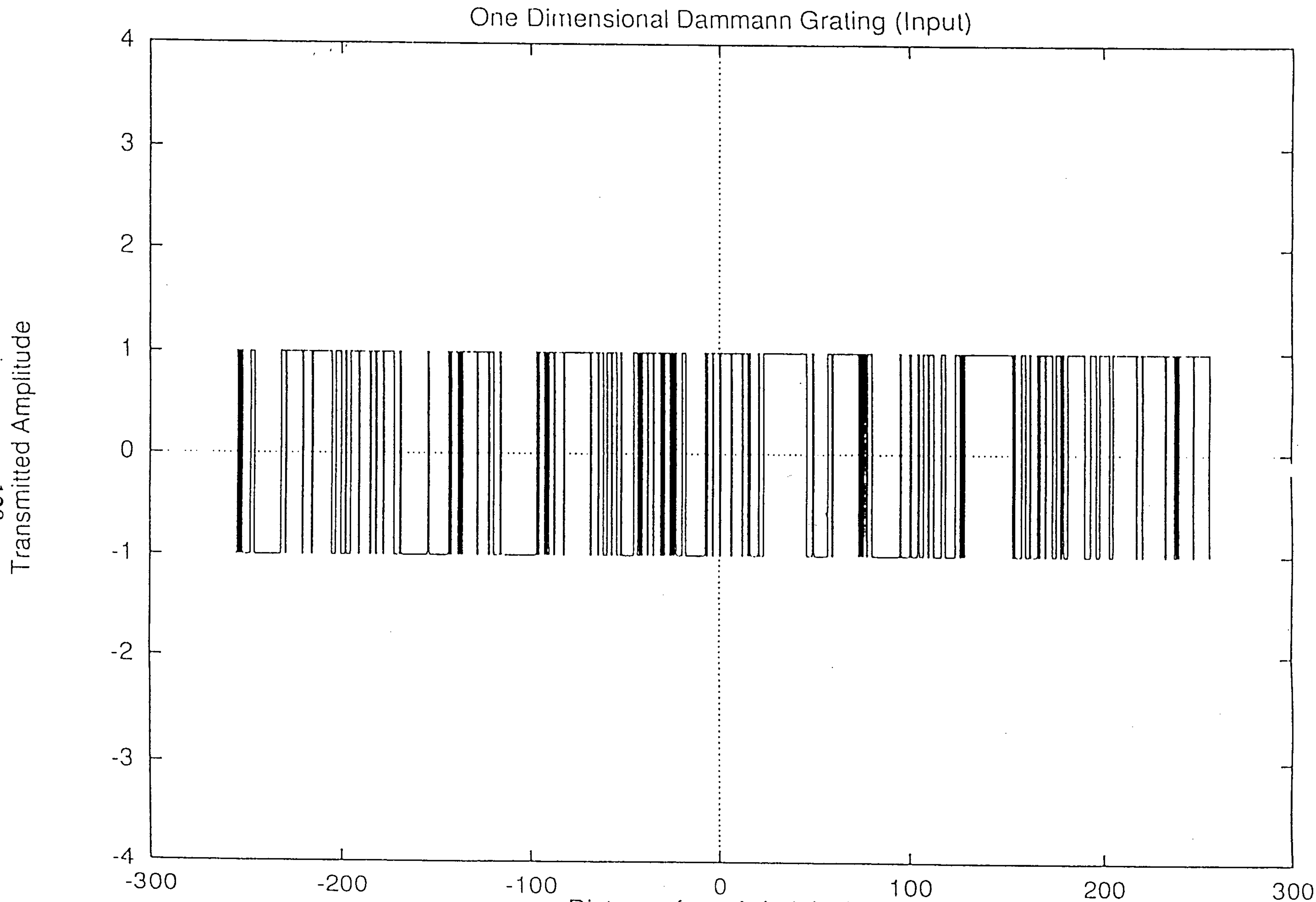


Fig. 4.68 17x17 High Mutation Rate One Dimensional Dammann Grating

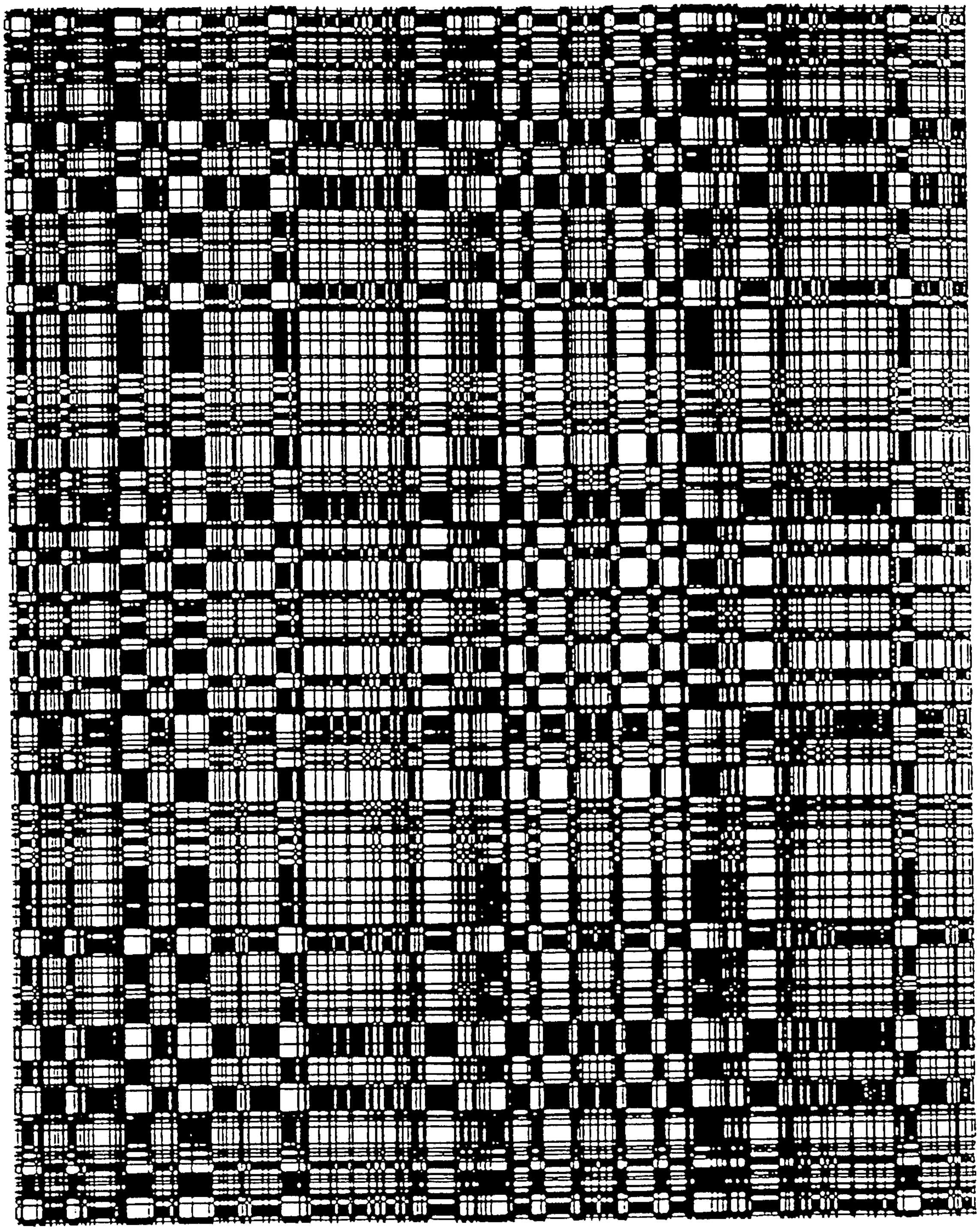


Fig. 4.69 17x17 High Mutation Rate Dammann Grating Cell

01111101 1110110 10000100 1111111 1111111
1111110 00100000 0011011 1111111 11010101
10000000 00000001 00001000 10011011 00001100
00011010 1111111 1111111 1111111 01000110
01000101 10111001 10100111 1111100 01110011
11100111 1111111 10110111 1111111 01111010
1111110 1111111 01010000 11000000 00000000
11011111 11101111 0111111 11100111 00100111
10111110 11011101 11110001 00000000 00000010
00000000 01011101 0111111 0111101 10001000
00000000 00000001 01101011 01111011 11111111
11101110 11001101 10010000 00110101 10110111
01011010 10001100 00000000 10110111 1.8217e-03 2978 297829

Fig. 4.70 17x17 High Mutation Rate Objective Function Value

Table 4.9 17x17 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0) (One Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.75	"	9.4250×10^{-4}
100	0.75	"	9.4148×10^{-4}
200	0.75	"	8.8443×10^{-4}
50	0.50	"	8.9964×10^{-4}
100	0.50	"	8.9234×10^{-4}
200	0.50	"	8.9334×10^{-4}
50	0.60	"	8.8650×10^{-4}
100	0.60	"	9.3510×10^{-4}
200	0.60	"	8.6172×10^{-4}

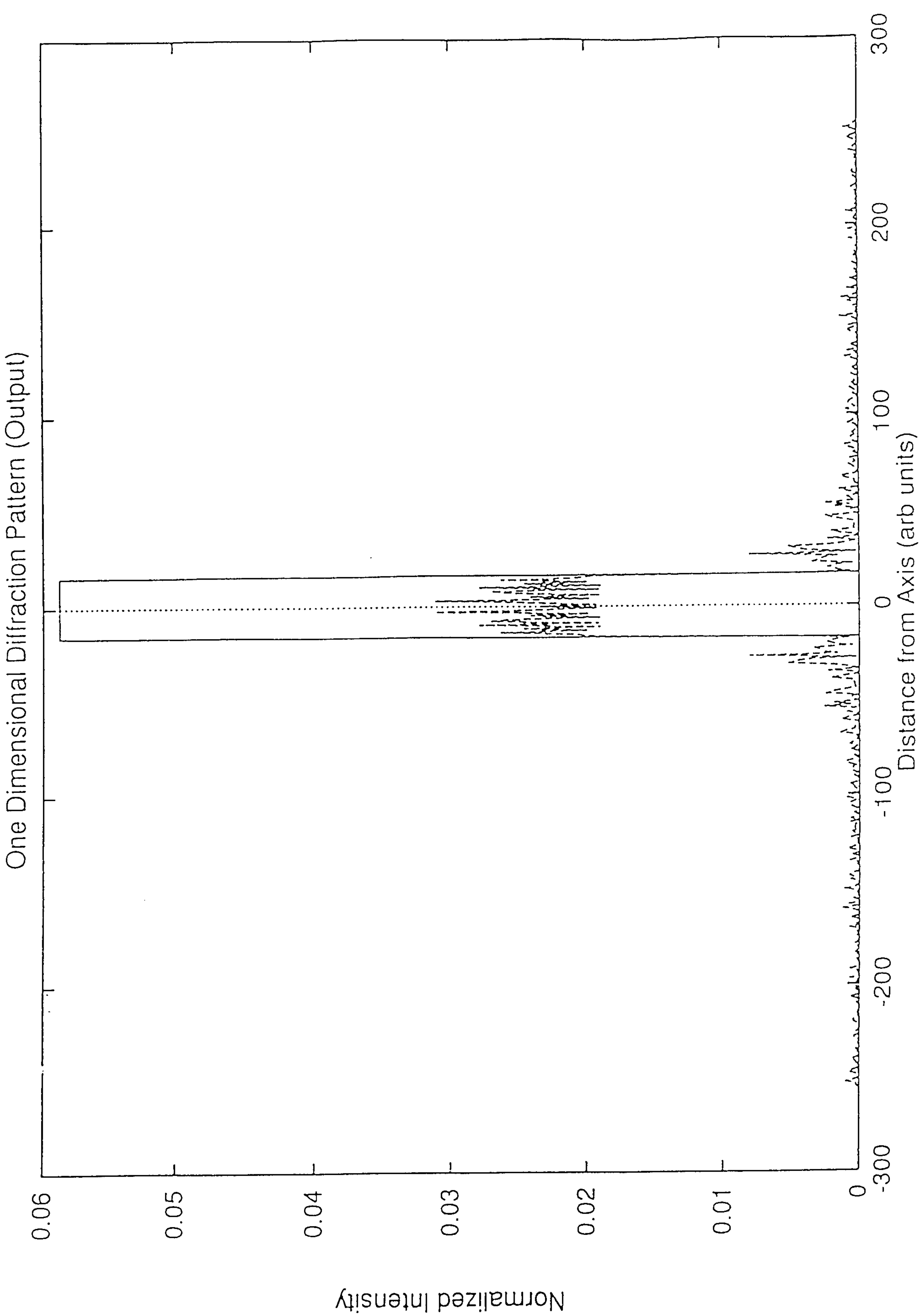


Fig. 4.71 17x17 Adaptive Mutation One Dimensional Diffraction Pattern

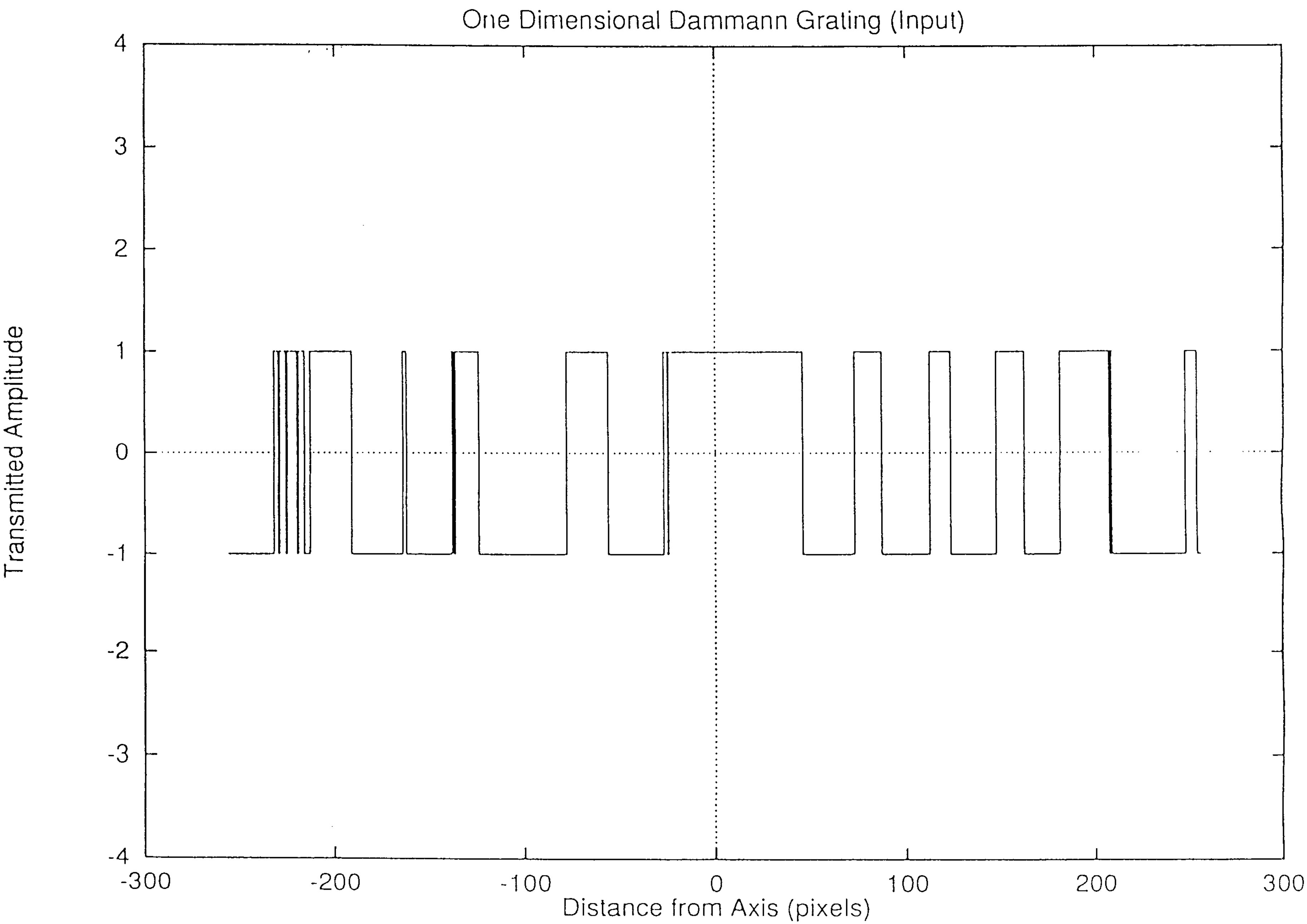


Fig. 4.72 17x17 Adaptive Mutation One Dimensional Dammann Grating

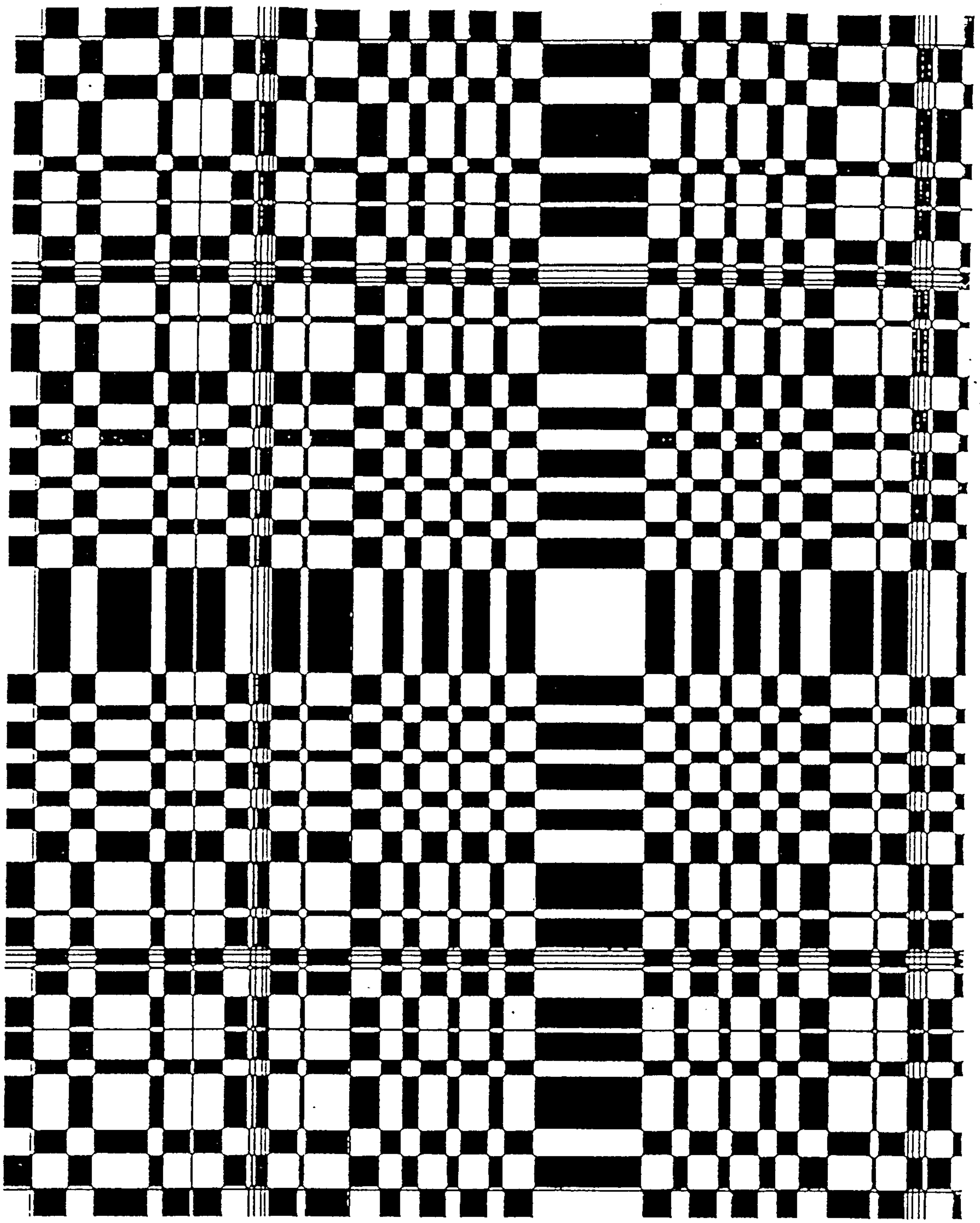


Fig. 4.73 17x17 Adaptive Mutation Dammann Grating Cell

11111111 11111111 11111111 11111111 11111111
11111100 00000000 00000000 00000000 01111111
11111110 00000000 00000000 00000000 11111111
11100000 00000000 00000000 00011111 11111111
11000000 00000000 00000111 11111111 11111111
11111110 10000000 00000000 00000000 00000000
00000000 11111100 00000000 00000000 00000000
11011101 11110111 00011111 11111111 11111111
10000000 00000000 00000000 00001100 00000000
00000000 00000010 11111111 11110000 00000000
00000000 00000000 00000000 00000000 00111111
11111111 11111111 00000000 00000000 00000000
00000110 11111111 11111111 11111111 9.4250e-04 758 32220

Fig. 4.74 17x17 Adaptive Mutation Objective Function Value

4.55; 4.56, 4.57 and 4.58 are the results obtained from using high or 10 per cent mutation rate. The results for the adaptive mutation and two point crossover scheme are shown in table 4.7 and in figures 4.59, 4.60, 4.61, and 4.62 respectively. In the simulation applying one point crossover and standard mutation, the results are shown in table 4.8 and in figures 4.63, 4.64, 4.65, and 4.66 respectively. Also shown in figures 4.67, 4.68, 4.69 and 4.70 are the results obtained for the one point crossover mechanism and high mutation rate of 10%. Table 4.9 and figures 4.71, 4.72, 4.73, and 4.74 respectively, show the results obtained from one point crossover and adaptive mutation mechanism.

Careful analysis of the results obtained for this particular array Dammann Gratings, show that, in the case of the two point crossover, standard mutation (lower rate), the diffraction efficiency is about 40% and the diffraction pattern fairly resolved. In the case of the application of high mutation rate, the diffraction efficiency fell to about 10% and showed noisy diffraction pattern with low resolution as observed in figure 4.55, 4.56, and 4.57 respectively. The two point crossover, adaptive mutation mechanism, produced similar results as was obtained from two point crossover, low standard mutation rate. These could be verified from tables 4.6 and 4.7 respectively. Also from figures 4.50 and 4.59.

From the one point crossover and standard mutation mechanisms, the best result obtained came a population of 200, 75 per cent crossover rate, and 0.1 per cent mutation rate. The diffraction efficiency as observed from figure 4.63 is about 57% on the average and has fairly resolved diffraction pattern. However, figure 4.67 which shows the diffraction pattern of the high mutation rate, produced as expected

Table 4.10 33x33 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.70	0.001	2.5085×10^{-4}
100	0.70	0.001	2.6050×10^{-4}
200	0.70	0.001	2.7252×10^{-4}
50	0.50	0.10	5.7169×10^{-4}
100	0.50	0.10	5.7467×10^{-4}
200	0.50	0.10	5.6469×10^{-4}
50	0.60	0.001	2.5341×10^{-4}
100	0.60	0.001	2.5731×10^{-4}
200	0.60	0.001	2.5908×10^{-4}

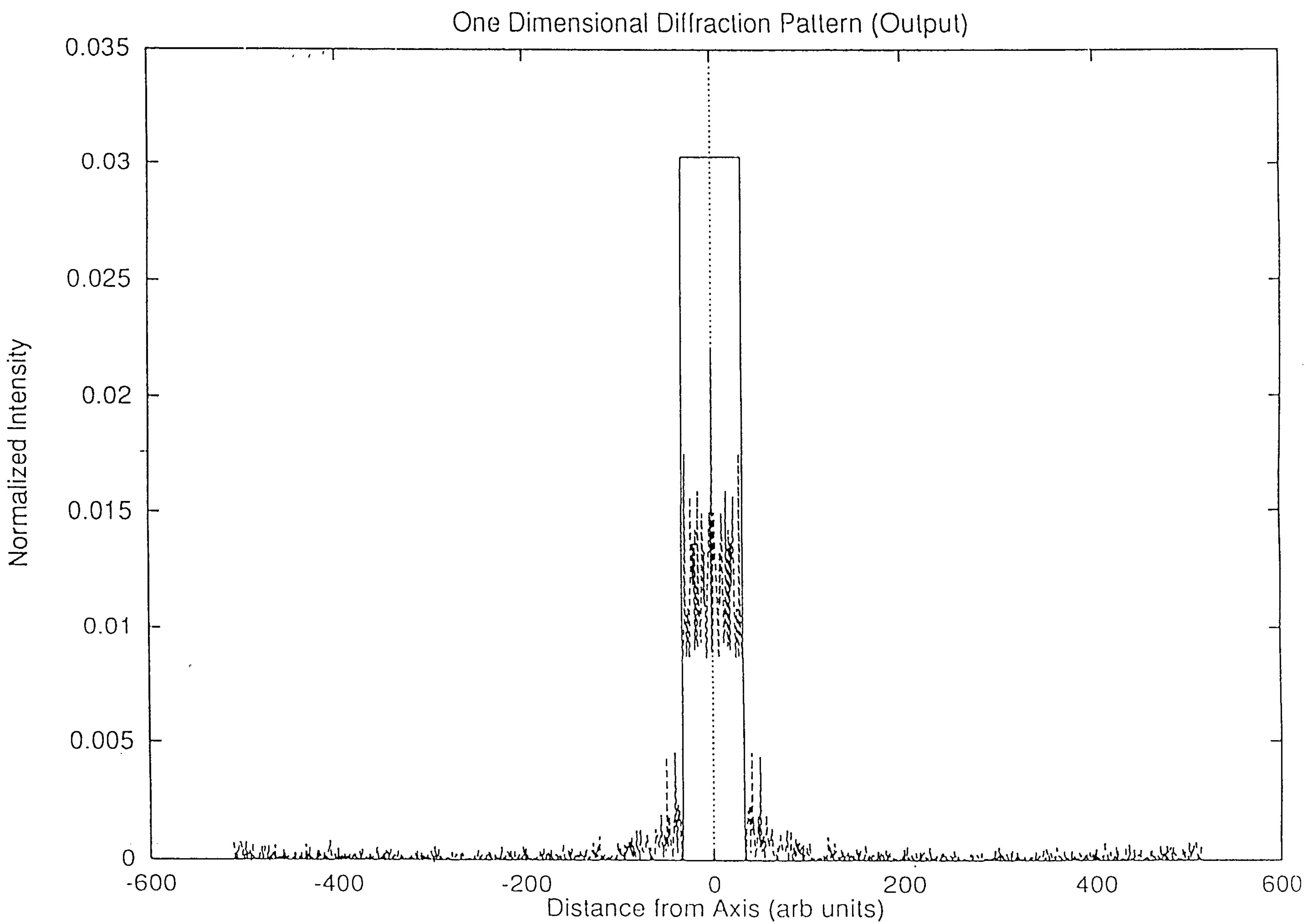


Fig. 4.75 33x33 One Dimensional Diffraction Pattern

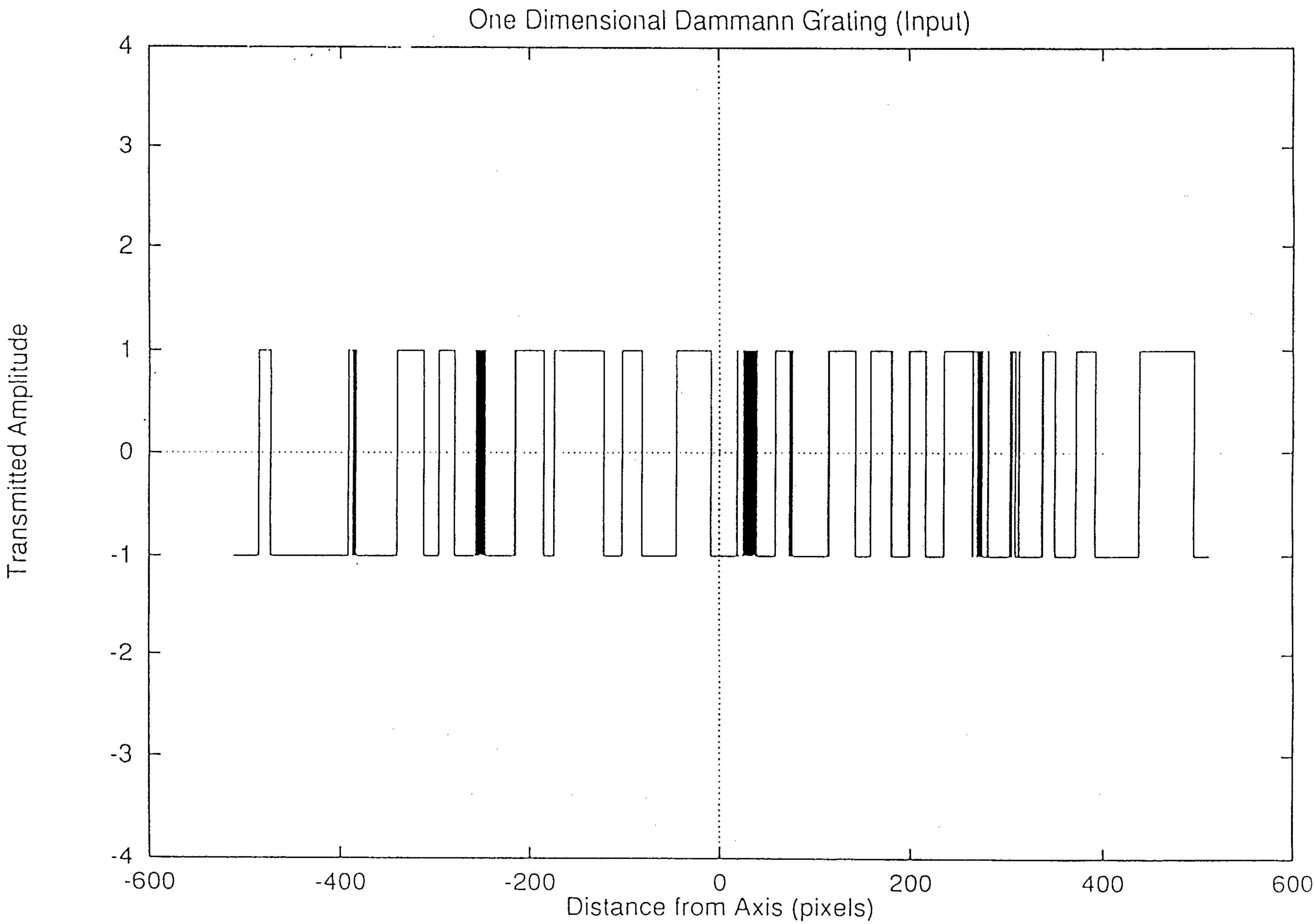


Fig. 4.76 33x33 One Dimensional Dammann Grating Structure

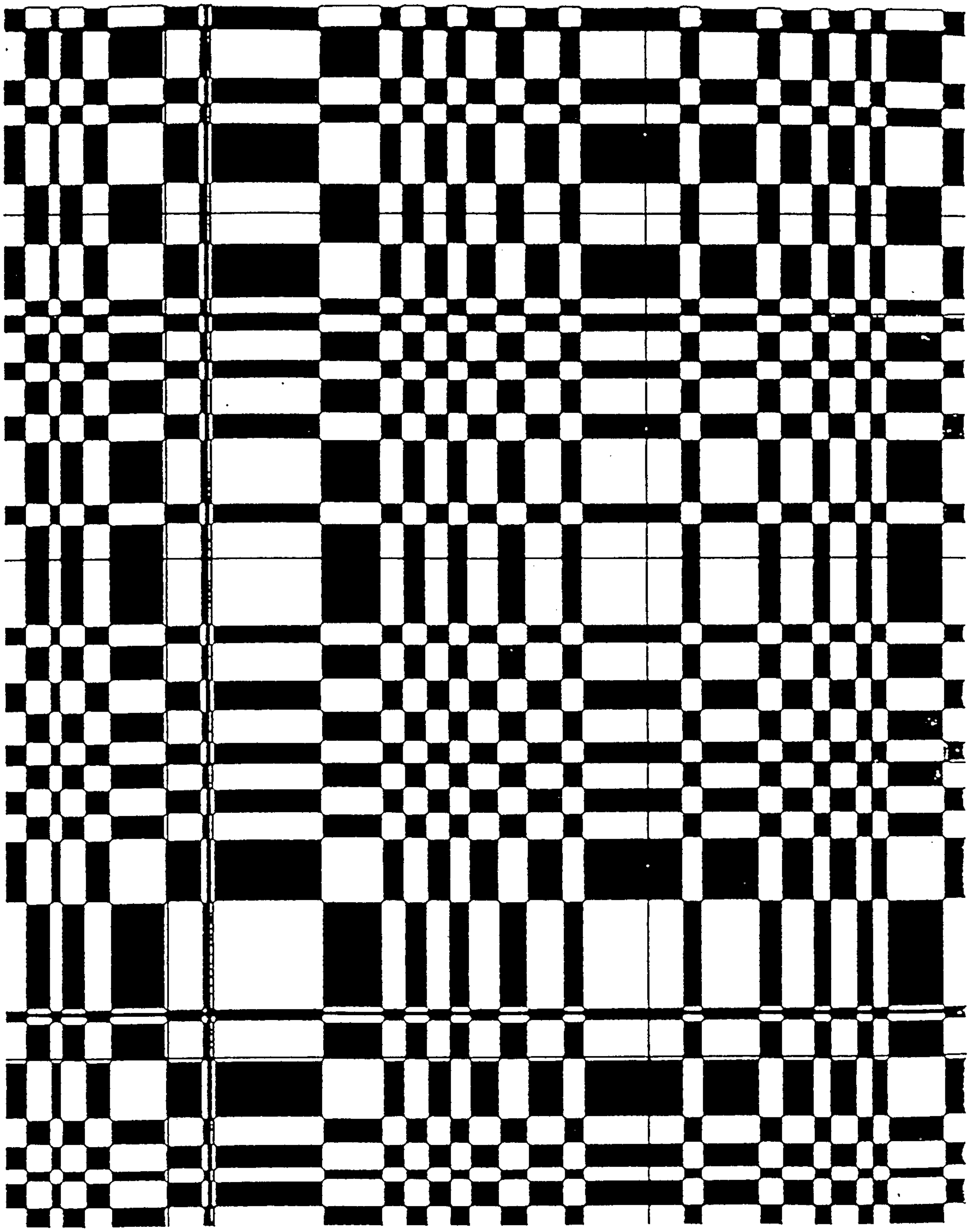


Fig. 4.77 33x33 Dammann Grating Cell

00000000 00000000 00100000 10101010 10011010 00000000
00000000 00111111 11111111 01010000 00000000 00000000
00000000 00000000 01011111 11111111 11111111 11111100
00000000 00000011 11111111 11111111 11101000 00000000
00000010 11111111 11111111 00000000 00000000 00011111
11111111 11111111 11111111 01111011 01100000 10000000
00000000 00000001 10111000 10000000 00000000 00000000
10111111 11111100 00000000 00000000 00001111 11111111
11111111 00000000 00000000 00000000 00000000 00000000
00000011 11111111 11111111 11111111 11111111 11111111
11111111 11111110 00000000 00000000 00000000 00000000
00000000 00011111 11111110 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000
00000000 01111010 10000000 00000000 00000000 00000000
00000000 00001111 11111111 11111111 11111111 00000000
00000000 11111111 11111111 10000000 00000000 00000001
01010110 11000000 00000000 00000000 00000000 10111111
11111111 11111111 11111110 00000000 00111111 11111111
11111111 11111111 11111111 11111111 11111110 00000000
00000000 00111111 11111111 11111110 00000000 00000000
00000000 00000000 00011111 11111111 11111111 11111111
11111110 00000000 2.6050e-04 2198 180425

Fig. 4.78 33x33 Objective Function Value

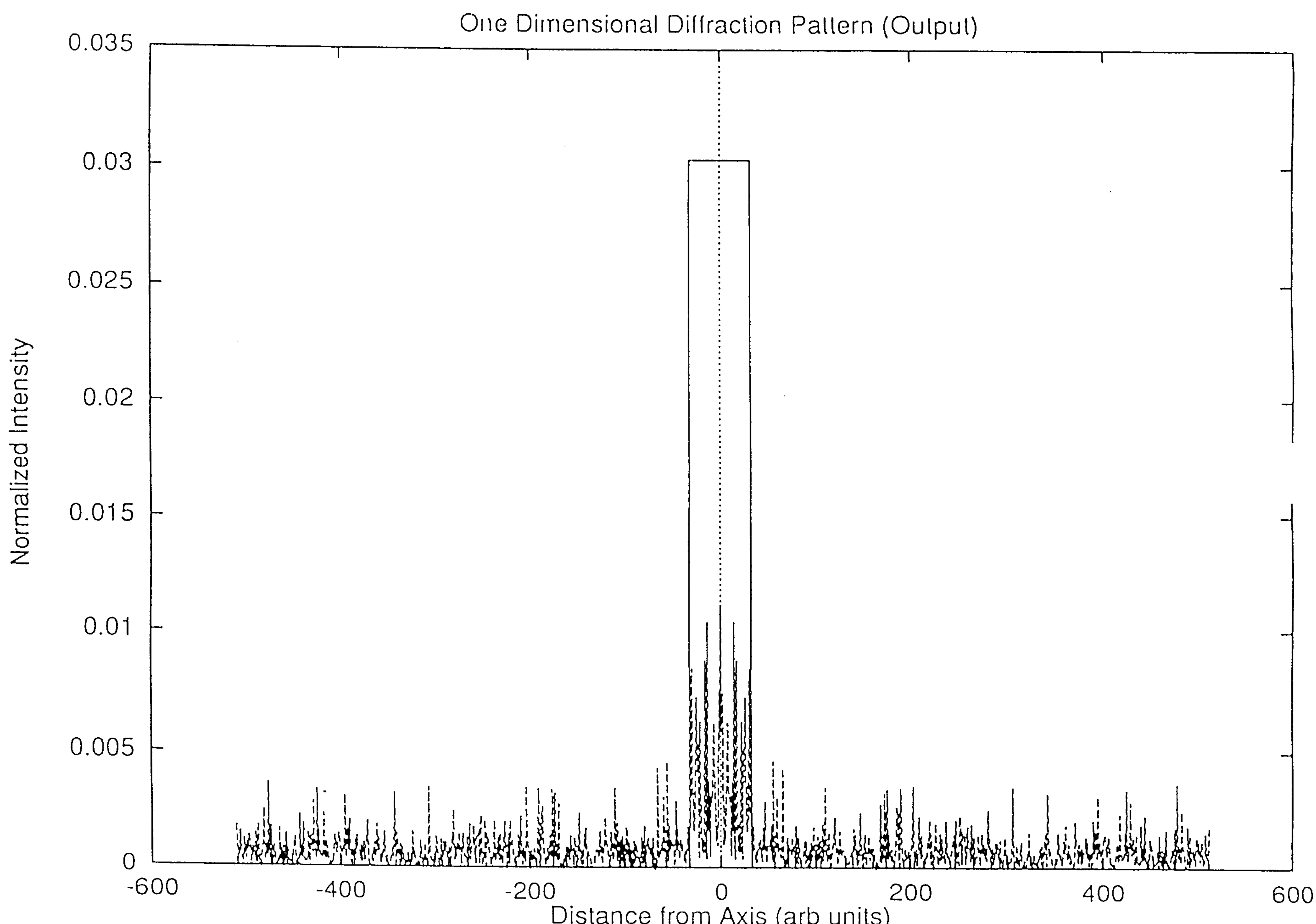


Fig. 4.79 33x33 One Dimensional Diffraction Pattern (High Mutation Rate)

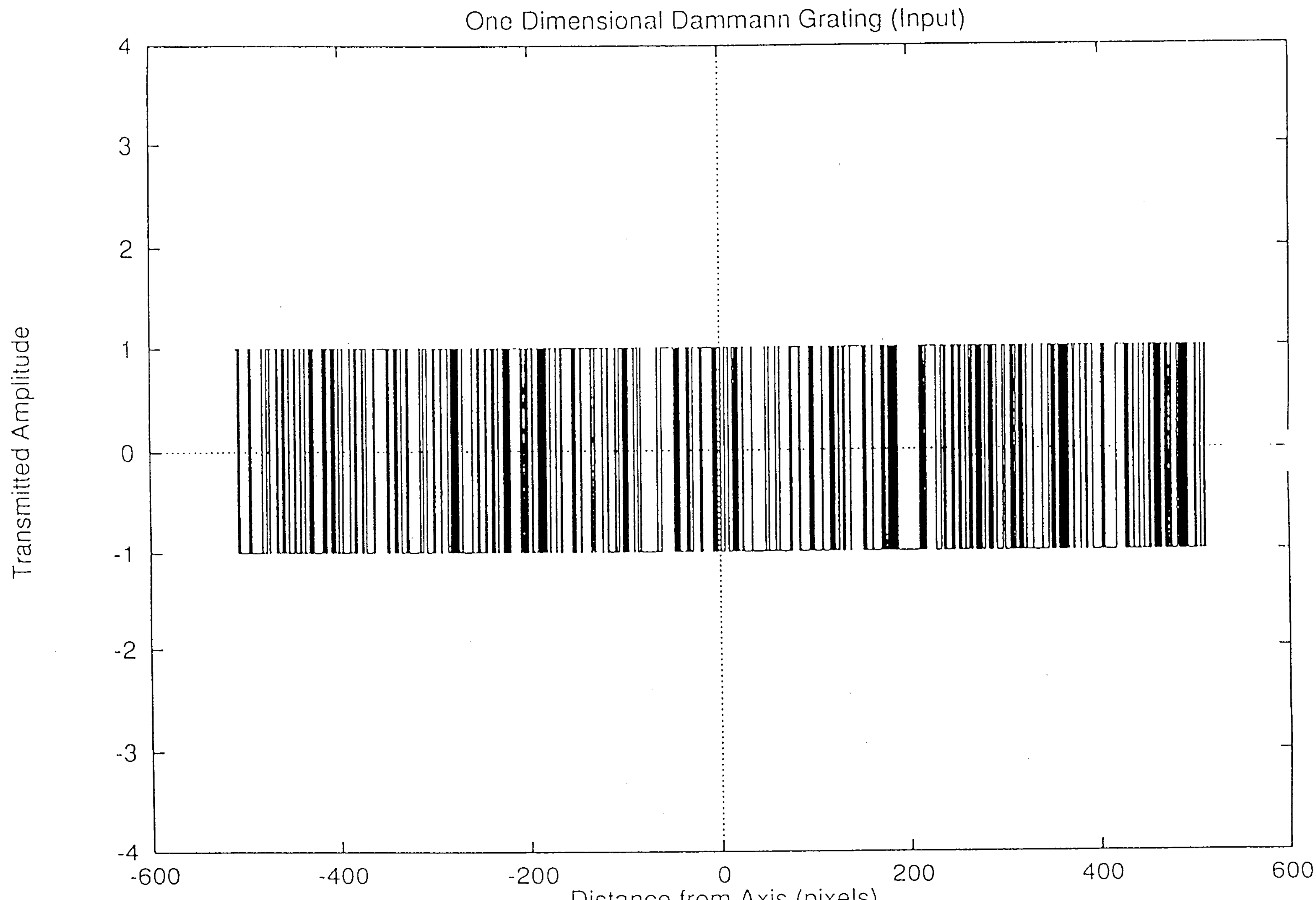


Fig. 4.80 33x33 One Dimensional Dammann Grating Structure (High Mutation Rate)

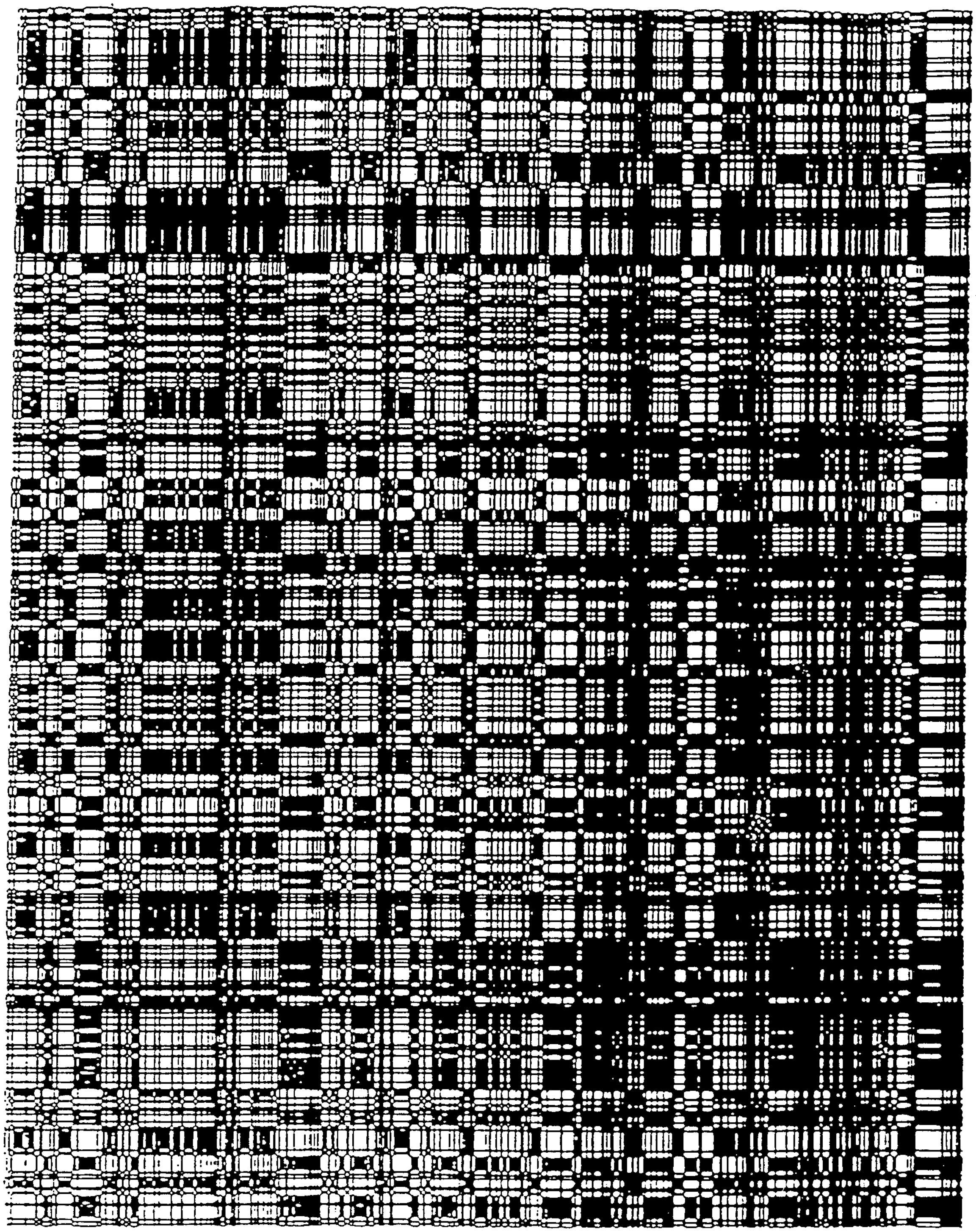


Fig. 4.81 33x33 Dammann Grating Cell (High Mutation Rate)

10000111 10000110 10110001 10000000 01000000 00000001
00010000 00100010 00000000 01101111 11101000 00000011
00110000 00001000 00001100 11000101 11010111 11011111
11111110 01100000 01000000 00101010 01010010 01010000
00000000 00000000 00101101 10111111 11110000 01110010
00001100 11101010 00100011 00111010 01010001 11001010
11100000 01110000 01010110 00110110 00100000 01000000
00010000 00111011 00111010 01010110 11110011 11110111
10101111 10000000 00101100 00000000 00111111 11101001
00001000 00100001 00001001 11010110 10111010 10011000
00110100 10010100 00001010 00100010 11110110 00000000
10100000 00000010 00011101 11111011 00011010 00100000
11000011 00010000 11001000 00000011 01011110 10110001
00010000 00010000 10011111 00111000 00001011 11111111
11001011 11010101 11000001 01000000 00000100 01111000
00011011 11100111 11100010 01001010 00100000 00001000
00101000 00100111 10110111 01111011 01100111 11111110
11001010 11101000 01001010 11011101 11110111 10011111
11111001 10111100 11111111 10111011 11111011 11010000
00110001 11001011 01111101 11000100 00000000 00000010
00011111 11111111 00110010 00000101 10001111 11101011
11111111 00110011 5.7467e-04 2961 296149

Fig. 4.82 33x33 Objective Function Value (High Mutation Rate)

very low the diffraction efficiency and low resolution. The input Dammann Grating of figure 4.68 also showed very low resolution and hence, a grating that could not be fabricated. The results from the one point crossover, adaptive mutation scheme, however showed similar results as obtained from low standard mutation two point crossover mechanism. Though, as observed from figure 4.71, the resolution of the diffraction pattern is not quite high. The diffraction efficiency is estimated to be about 47% with the best result of the simulaltion given by a population of 50, and 75 per cent crossover rate.

4.3 33 x 33 Amplitude Dammann Gratings

To devise this size amplitude Dammann Gratings, the GenesYs 1.0 package was also utilised with 1024 bit binary elements. This simulation was carried out with a one and two point crossover, standard, and adaptive mutation mechanisms. Due to the long bit string, the degree of computation difficulty increases. Time of computation also increases. For a single run on the Sun workstation, the time required to complete computation is in the order of 60 minutes for the low mutation and adaptive mutation shemes. The time increases with high standard mutation rate. All the simulations for this size grating were carried out with 300,000 trials. For the two point crossover and standard mutation, the results obtained are shown in table 4.10 and in figures 4.75, 4.76, 4.77, and 4.78 respectively for a population size of 100, 70 per cent crossover rate, and 0.1 per cent mutation rate. In this simulation, the diffraction pattern obtained was of low resolution and of about 30% diffraction efficiency. Also shown in figures 4.79, 4.80, 4.81 and 4.82 are

Table 4.11 33x33 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.70	"	2.6313×10^{-4}
100	0.70	"	2.3933×10^{-4}
200	0.70	"	2.6000×10^{-4}
50	0.50	"	2.4171×10^{-4}
100	0.50	"	2.5319×10^{-4}
200	0.50	"	2.4449×10^{-4}
50	0.60	"	2.4884×10^{-4}
100	0.60	"	2.4201×10^{-4}
200	0.60	"	2.3874×10^{-4}

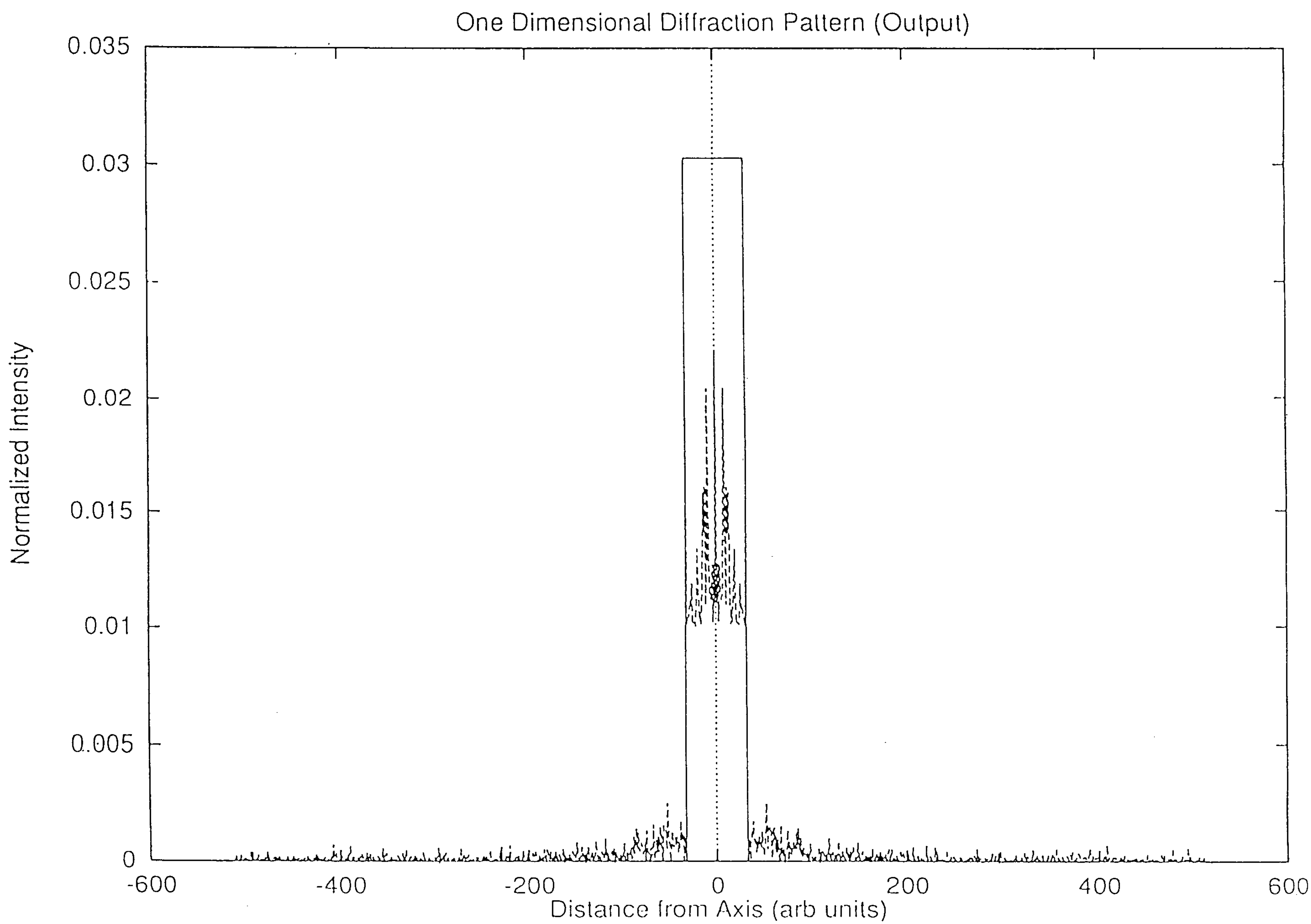


Fig. 4.83 33x33 One Dimensional Diffraction Pattern (Adaptive Mutation)

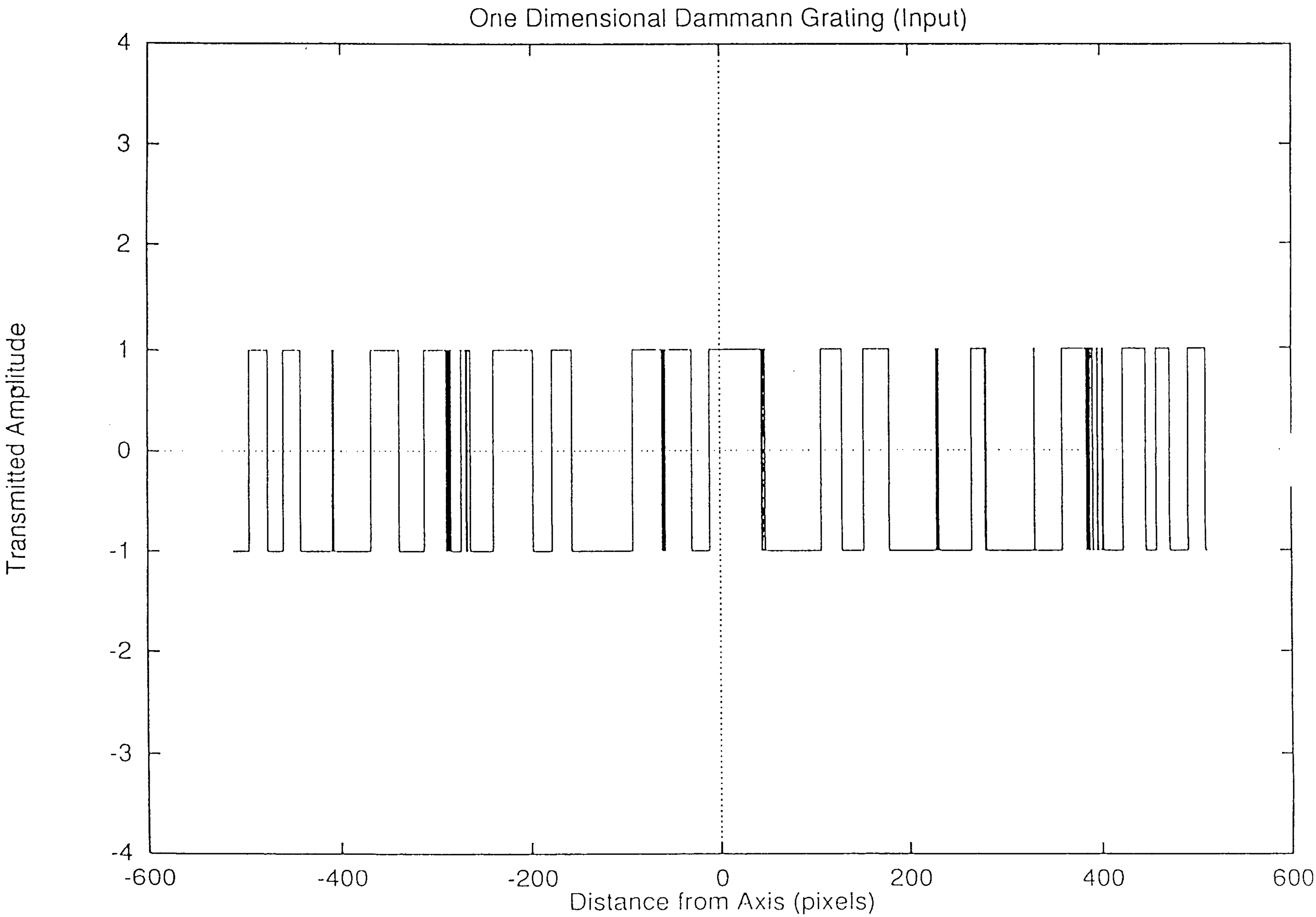


Fig. 4.84 33x33 One Dimensional Dammann Grating Structure (Adaptive Mutation)

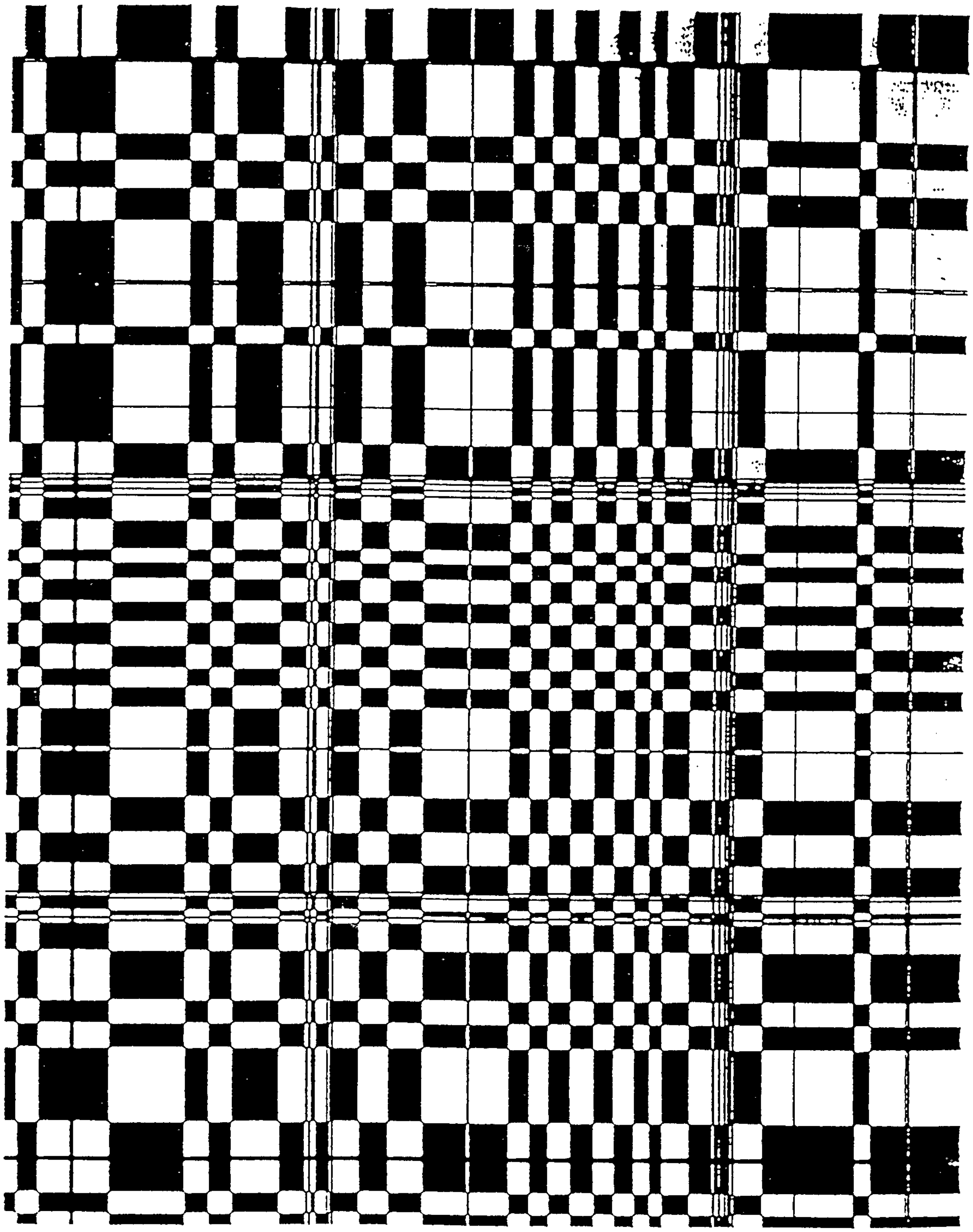


Fig. 4.85 33x33 Dammann Grating Cell (Adaptive Mutation)

11111111 11111111 11111111 11111111 11111111 11101001
00000000 00000000 00000000 00000000 00000000 00000000
00000000 00011111 11111111 11111111 10000000 00000000
00000000 11111111 11111111 11111111 11100000 00000000
00000000 00000000 00000000 00000000 00001001 00000000
00000000 00000000 00000000 01111111 11111110 10000000
00000000 00000000 00000000 00000000 00000000 00100000
00000000 00000000 00000001 11111111 11111111 11111111
01101001 00001000 01100000 00000000 00000001 11111111
11111111 11111110 00000000 00111111 11111111 00000000
00000000 00011111 11111111 11111000 00000000 00000000
01111111 11111111 11110000 00000000 00001111 11111111
11111100 00000000 00000000 00000000 00000001 10000000
00000000 00000000 00000000 00000000 11111111 11111111
11111111 11111100 00000000 00000000 00000000 11111111
11111111 11111110 11001000 00000001 11110011 10000000
00000000 00000000 01111111 11111111 11111111 11111111
11111111 11100000 00000000 00000001 11111111 11111111
11110000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00001111 11111111 11111111 11111111
11010101 11111111 11111111 11111111 11000000 00000000
00000111 11111111 2.3933e-04 3132 268126

Fig. 4.86 33x33 Objective Function Value(Adaptive Mutation)

Table 4.12 33x33 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.70	0.001	2.4093×10^{-4}
100	0.70	0.001	2.5149×10^{-4}
200	0.70	0.001	2.3368×10^{-4}
50	0.50	0.10	5.6223×10^{-4}
100	0.50	0.10	5.7695×10^{-4}
200	0.50	0.10	5.8288×10^{-4}
50	0.60	0.001	2.5689×10^{-4}
100	0.60	0.001	2.6387×10^{-4}
200	0.60	0.001	2.5797×10^{-4}

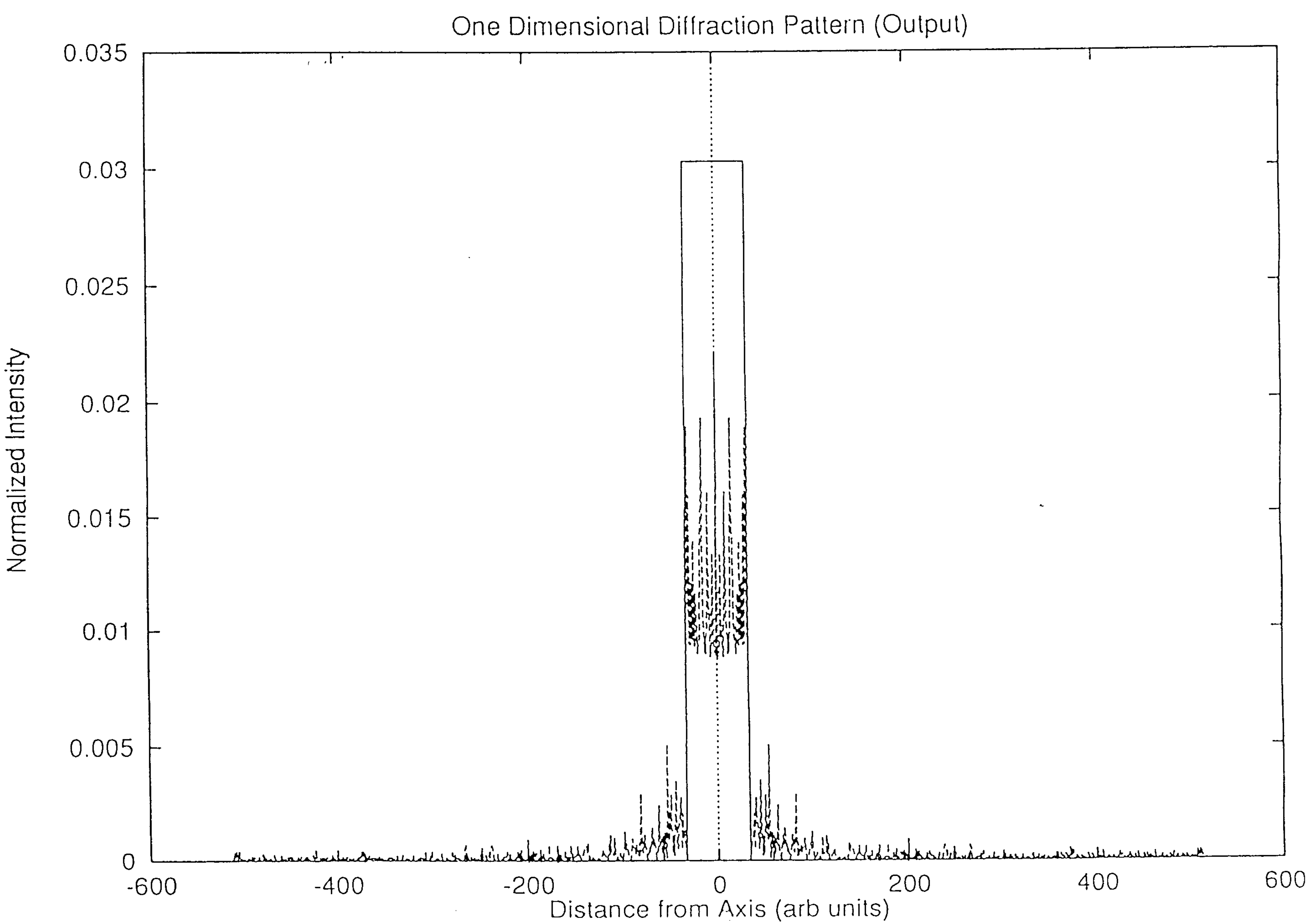


Fig. 4.87 33x33 One Dimensional Diffraction Pattern

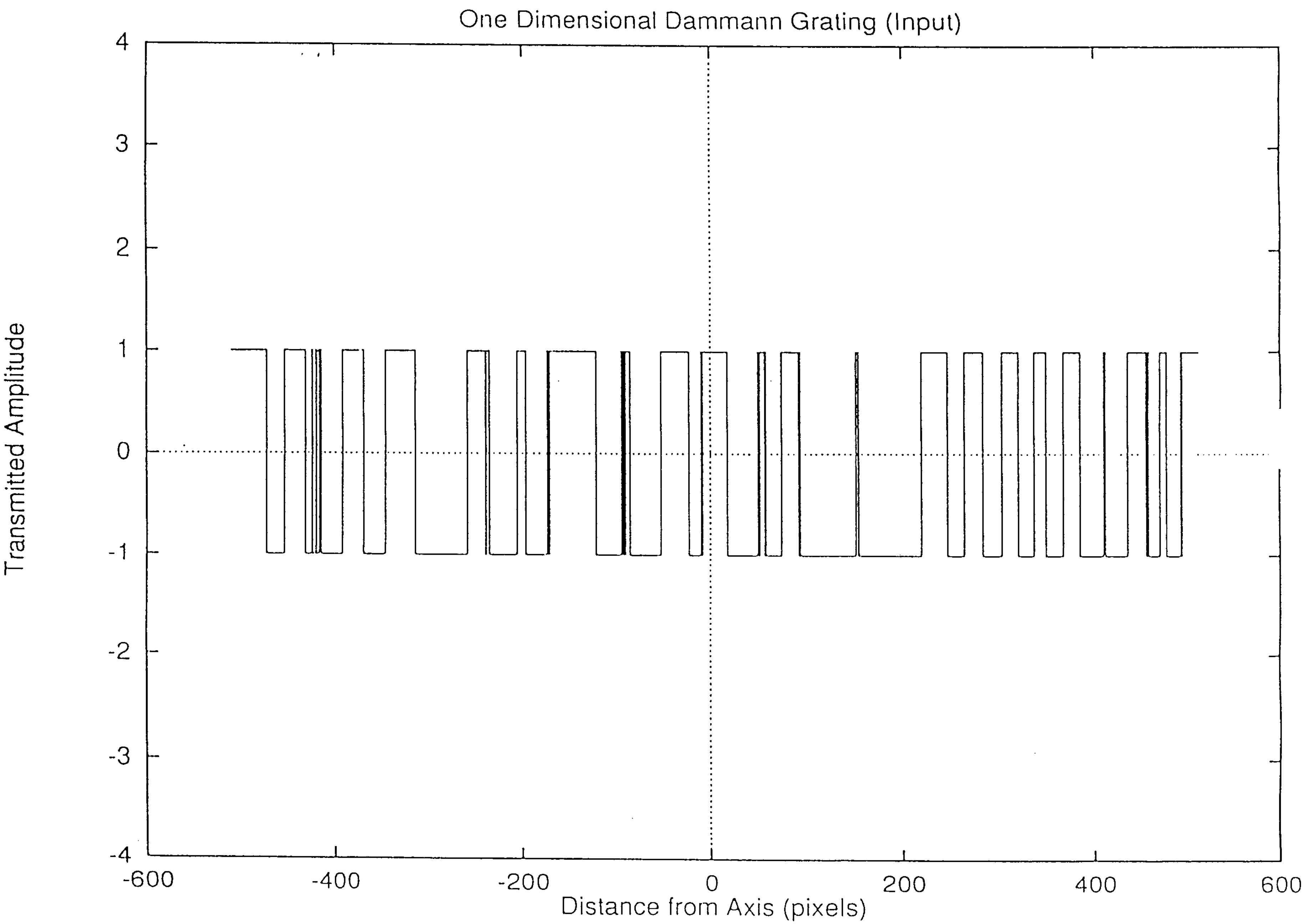


Fig. 4.88 33x33 One Dimensional Dammann Grating Structure

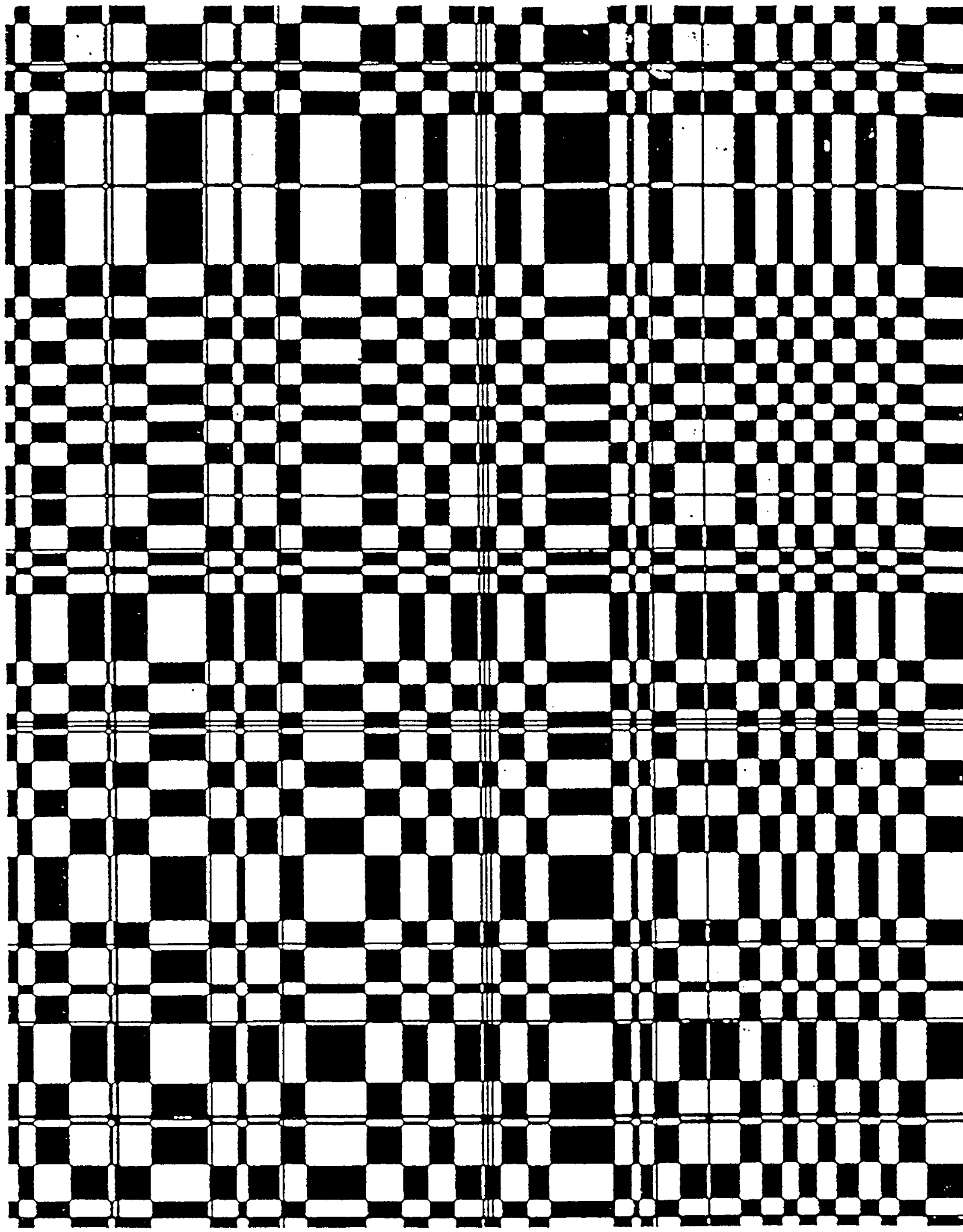
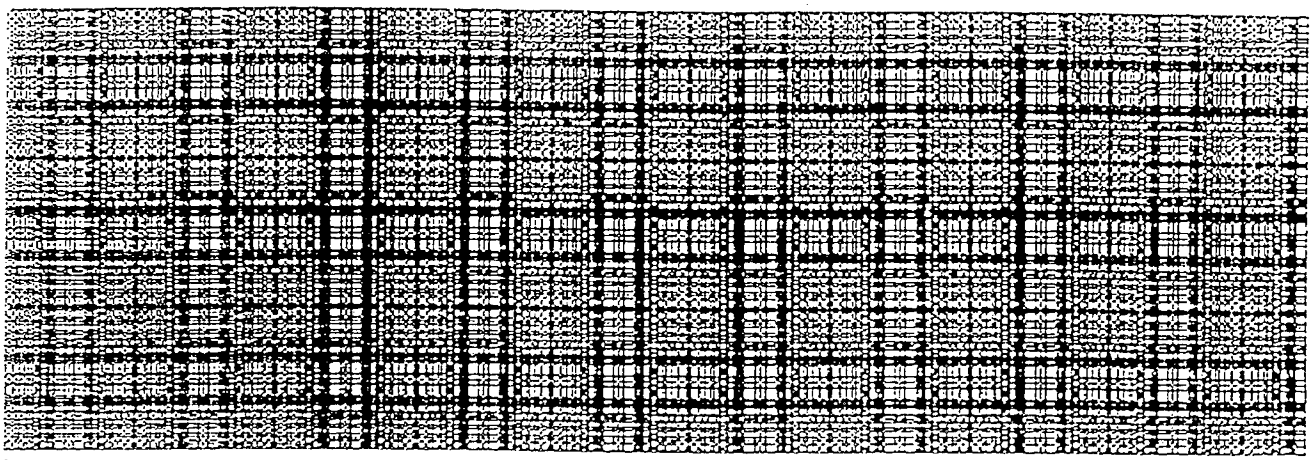


Fig. 4.89 33x33 Dammann Grating Cell



7.46 2048

Fig. 4.90 33x33 Fabricated Amplitude Dammann Grating Device

11111111 11111111 11000000 00000000 00000000 00000000
00110111 10100000 00000000 00011111 11111111 11111010
00000000 00000000 00000000 00000000 00000000 00000000
00000000 11100000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00011111 11111111 11111111
11111100 00000000 00000001 11111111 11111111 11000000
00000000 00000011 11111111 11111110 00000000 00000000
11111111 11111000 00000000 00000001 11111111 11111111
10000000 00000000 00000000 00110000 00000000 00000000
00001111 11111111 11111110 11000000 00000011 11111000
00000000 00001011 11111111 11111111 11111111 11111111
11111111 11111111 11111100 00000000 00000000 01111111
11111111 11111111 00000001 00010001 10000000 00000000
00000000 01111111 11111111 11111111 00000000 00000000
00000000 11111111 11111111 11111111 11111111 00000000
00000000 00000000 00000000 00000000 00000000 00000001
11111111 11111111 11011100 00000000 00000000 00000000
00011111 11110000 00000000 00000000 00100111 11111111
11111111 11111111 11111111 11111111 11111100 00000000
00000000 00000000 01010011 11000000 00000000 00000000
00000000 00011111 11111111 11111111 11111111 10000000
00000010 11111111 2.6387e-04 2810 213941

Fig. 4.91 33x33 Objective Function Value

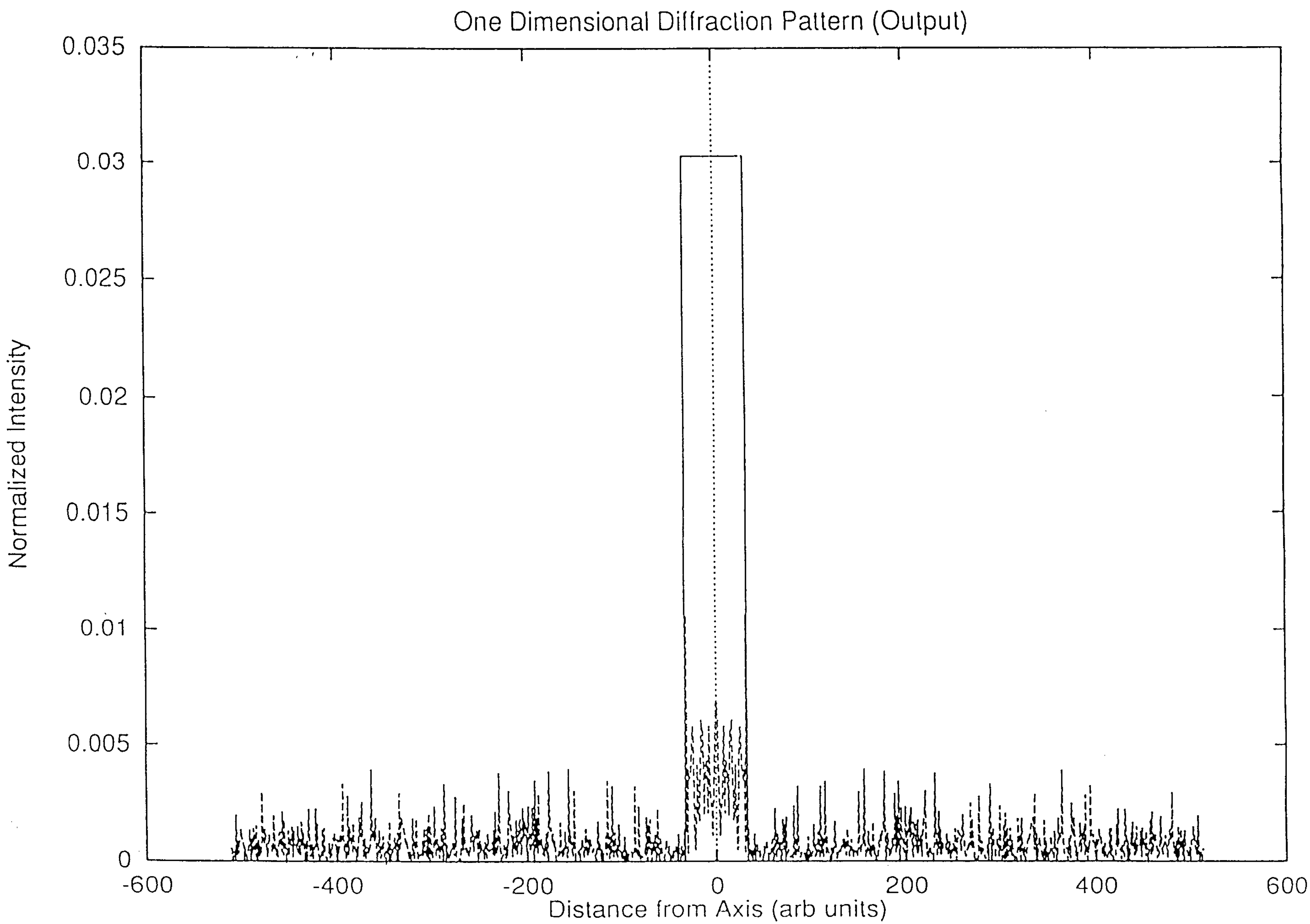


Fig. 4.92 33x33 One Dimensional Diffraction Pattern (High Mutation Rate)

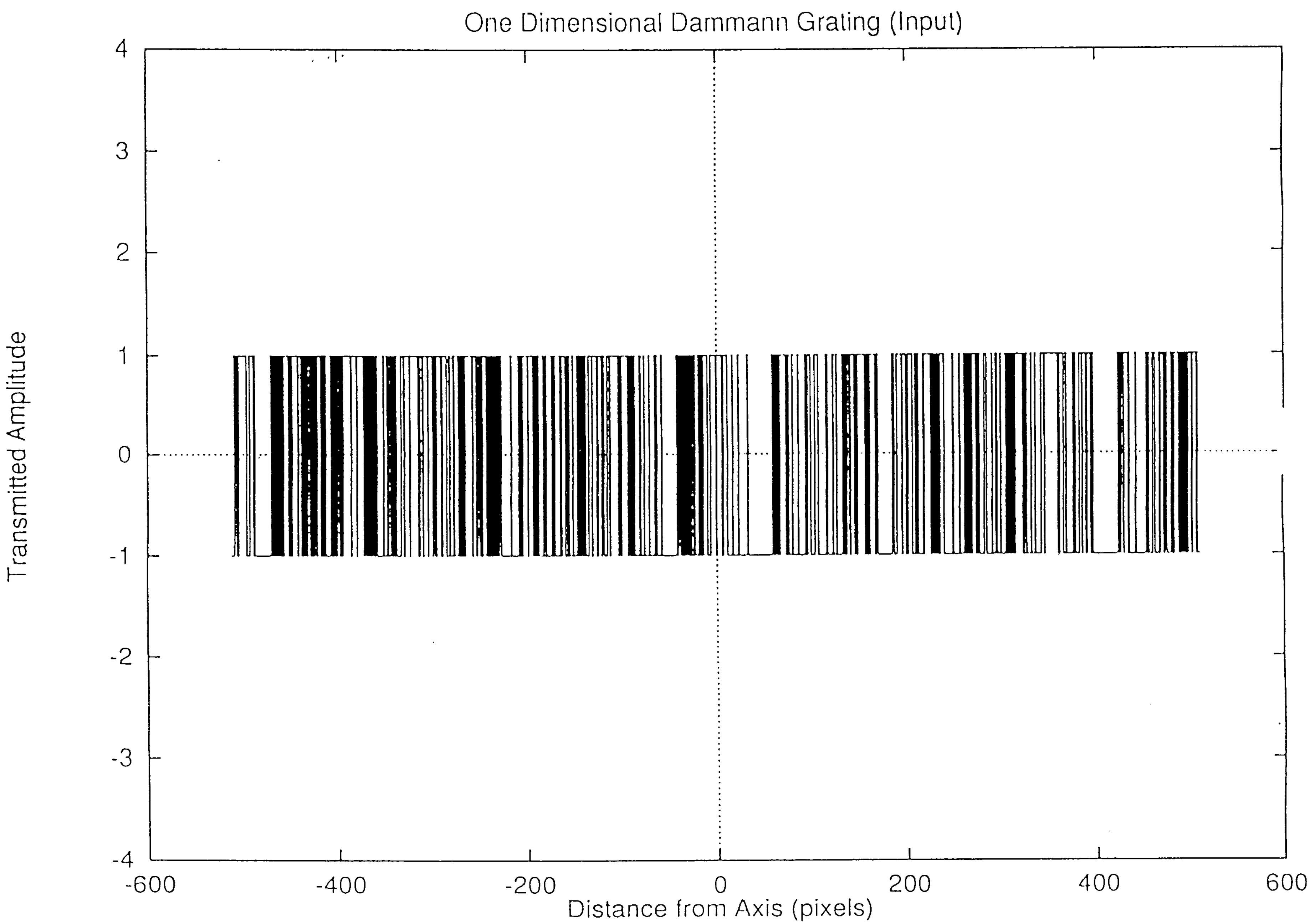


Fig. 4.93 33x33 One Dimensional Dammann Grating Structure (High Mutation Rate)

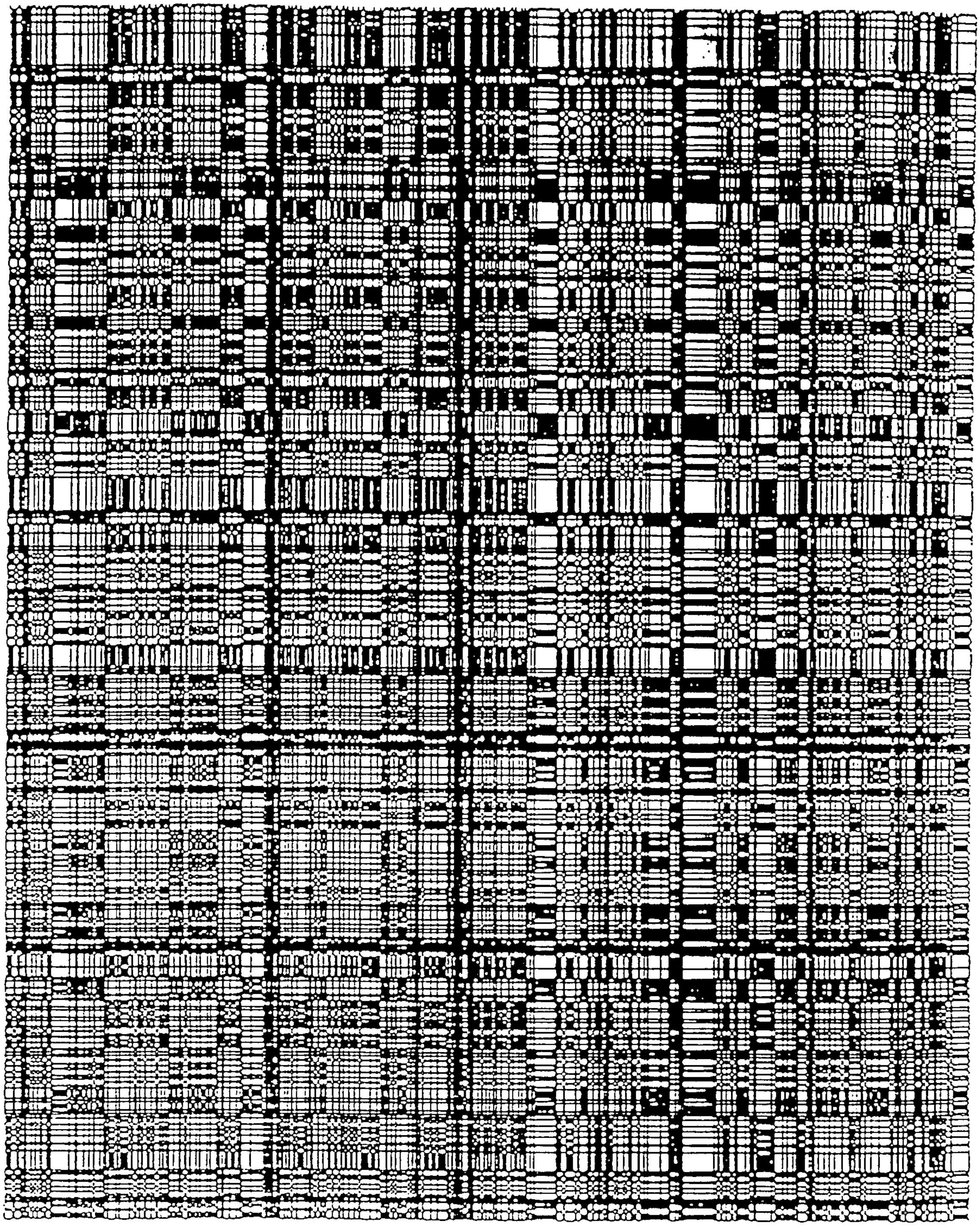


Fig. 4.94 33x33 Dammann Grating Cell (High Mutation Rate)

11111001 11100000 10000010 00000000 10000000 00000000
00000000 00100110 10011111 01010001 00000100 00000100
11100001 11100000 00011000 00100010 00001100 10110011
10110111 11110110 10111110 11000000 00000000 01100011
11101110 10111100 01101000 01011111 10011001 01001111
00000000 01000000 10000110 01011011 10110100 00111000
00100011 00010000 10011010 10011111 11010010 00010001
00000011 10111111 11111110 10111000 10000001 01100011
10111001 11001000 00000000 00000000 00000001 01011011
11010000 00100000 00000100 10001110 00010111 00110000
11001111 01010011 01011100 11101000 00011100 11111111
00011110 10000000 00000000 00100100 10110110 11110110
01111101 11010011 01101100 11011101 01010000 10101010
11001011 11111101 11110111 11100110 10101100 11000000
10001100 11001010 00011101 11110111 11111001 11011110
11110110 01111001 11100010 00110001 10101101 11111101
11001101 10111010 10010101 01011000 00000001 00000001
01101111 10111101 01101000 01001000 00100100 00100111
00101000 10001011 01101100 01110111 10011101 10001110
11111111 01101111 11011010 01000011 00010000 10000010
10000100 00000000 00000101 10100101 10011001 10001010
10111100 01111110 5.7695e-04 2841 284154

Fig. 4.95 33x33 Objective Function Value (High Mutation Rate)

Table 4.13 33x33 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.70	"	2.7922×10^{-4}
100	0.70	"	2.7449×10^{-4}
200	0.70	"	2.5830×10^{-4}
50	0.50	"	2.5868×10^{-4}
100	0.50	"	2.6117×10^{-4}
200	0.50	"	2.5506×10^{-4}
50	0.60	"	2.6912×10^{-4}
100	0.60	"	2.5117×10^{-4}
200	0.60	"	2.8547×10^{-4}

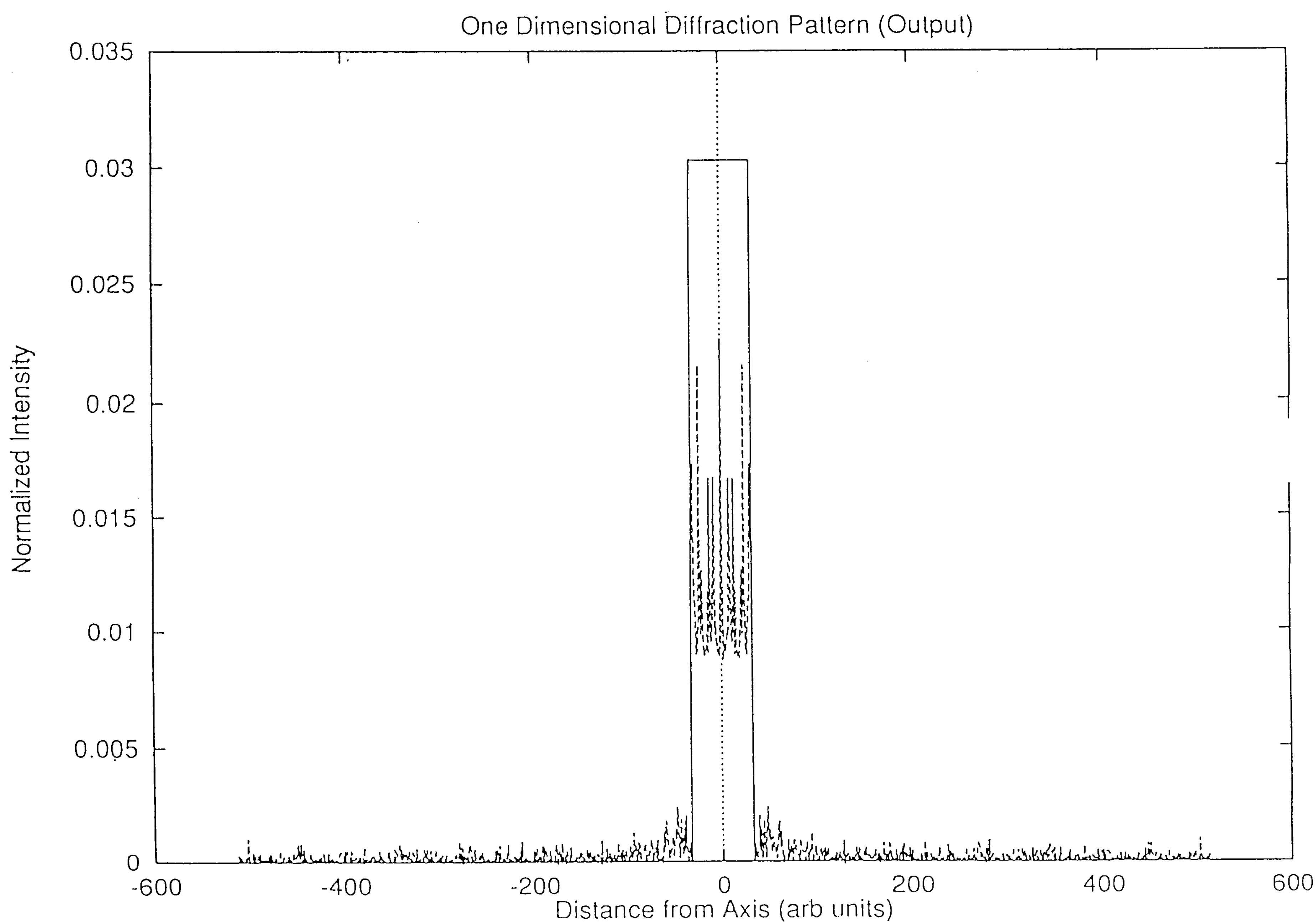


Fig. 4.96 33x33 One Dimensional Diffraction Pattern (Adaptive Mutation)

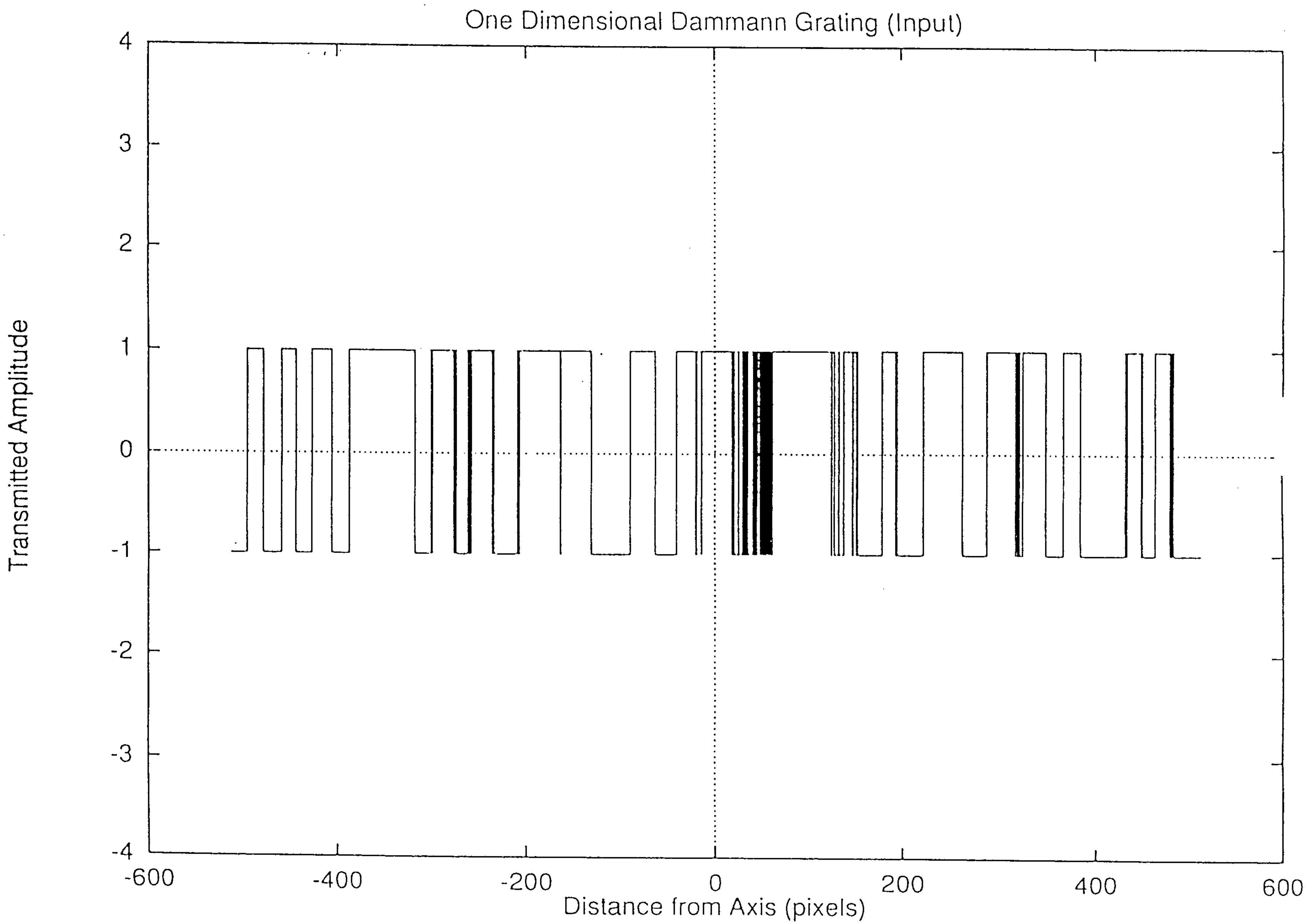


Fig. 4.97 33x33 One Dimensional Dammann Grating Structure (Adaptive Mutation)

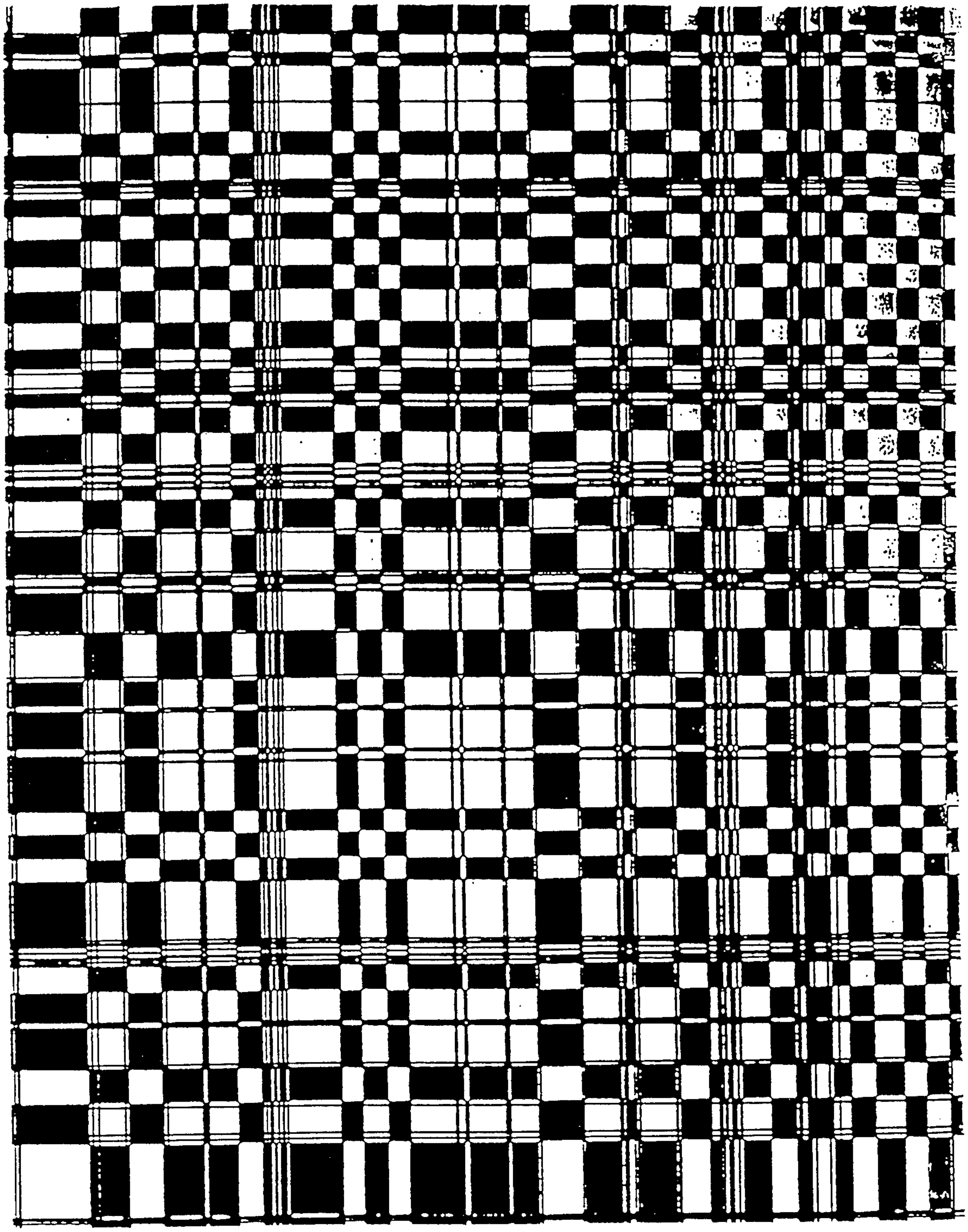


Fig. 4.98 33x33 Dammann Grating Cell (Adaptive Mutation)

11111111 11111111 11111101 10000000 00000000 11101111
11100000 00000000 00000000 00000000 10000000 00000000
00000000 01111111 11111111 11101000 00000000 00000000
11011011 10111101 00000000 00010111 11111111 11111110
10000000 00000000 00000011 11111111 11111111 11000000
00000000 00000000 00001111 11111111 11111111 10100000
00001000 00111011 11111111 10111010 10000101 00111111
11111111 11111111 00000000 00000000 00000000 00111001
11010001 11001010 00000010 11111111 11111111 11111111
00110000 00000000 00000000 00000000 00001001 10111110
10011010 00000000 00000000 00000000 00000001 10111111
11111111 11111111 11111111 11111011 00000000 00000000
00000001 10010000 00000000 00000000 00000000 00011110
00001010 00000000 00000000 00000000 00000000 00000010
11111111 11111111 11000000 00000000 00000000 11111111
11111111 11010000 00000000 00000000 00000000 00000000
00000000 00001001 01001100 01101001 00101111 11111111
11111111 10000000 00000000 00000000 00101101 00000000
00000000 00000000 00000010 01011111 11111111 11111111
11111110 00100000 00000000 00000000 01000010 00111111
11111111 11111111 11111111 11111111 11111111 11111111
11111110 10110111 2.8547e-04 2227 288536

Fig. 4.99 33x33 Objective Function Value(Adaptive Mutation) the high mutation rate results. As can observed from the diffraction pattern of figure 4.79, the diffraction efficiency is of the order of 9%. The resolution of the pattern is as expected, very low. The input grating of figure 4.80 also showed very low resolution. This undesirable results are attributed to the high mutation rate used for the simulation. The grating cell of figure 4.81 also is of very low resolution. The implication of this, is that the grating could not be useful if fabricated either as amplitude grating or phase grating. For the adaptive mutation and two point crossover case, the results are as shown in table 4.11 and in figures 4.83, 4.84, 4.85 and 4.86 respectively. The diffraction efficiency is estimated to be about 40% with low resolution of the diffraction pattern. The input grating as inferred from figure 4.84, is fairly resolved as well as the grating cell. The results obtained for the one point crossover mechanism and for standard mutation rate are shown in figures 4.87, 4.88, 4.89, 4.90 and 4.91. Figure 4.90, shows the fabricated amplitude 33x33 Dammann Grating with 2540 dpi (dots per inch) resolution. The diffraction efficiency is about 40% with fairly resolved diffraction pattern. The input grating as observed in figure 4.88 is also fairly resolved. Shown in figures 4.92, 4.93, 4.94, and 4.95 are the results of high mutation rate. From figure 4.92, it could be estimated that the diffraction efficiency is about 7%, and resolution of the diffraction pattern is very low. The resolution of the input grating of figure 4.93 also is of low value. The Dammann Grating cell obtained in this simulation also is of very low resolution as is observed in figure 4.94. For the one point crossover and adaptive mutation scheme, the results obtained from this simulation are shown in table 4.13 and in figures 4.96, 4.97,

Table 4.14 65x65 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.70	0.001	8.4808×10^{-5}
100	0.70	0.001	8.9701×10^{-5}
200	0.70	0.001	1.0105×10^{-5}
50	0.50	0.10	1.6247×10^{-4}
100	0.50	0.10	1.6025×10^{-4}
200	0.50	0.10	1.6202×10^{-4}
50	0.60	0.001	8.0708×10^{-5}
100	0.60	0.001	8.6959×10^{-5}
200	0.60	0.001	1.0082×10^{-4}

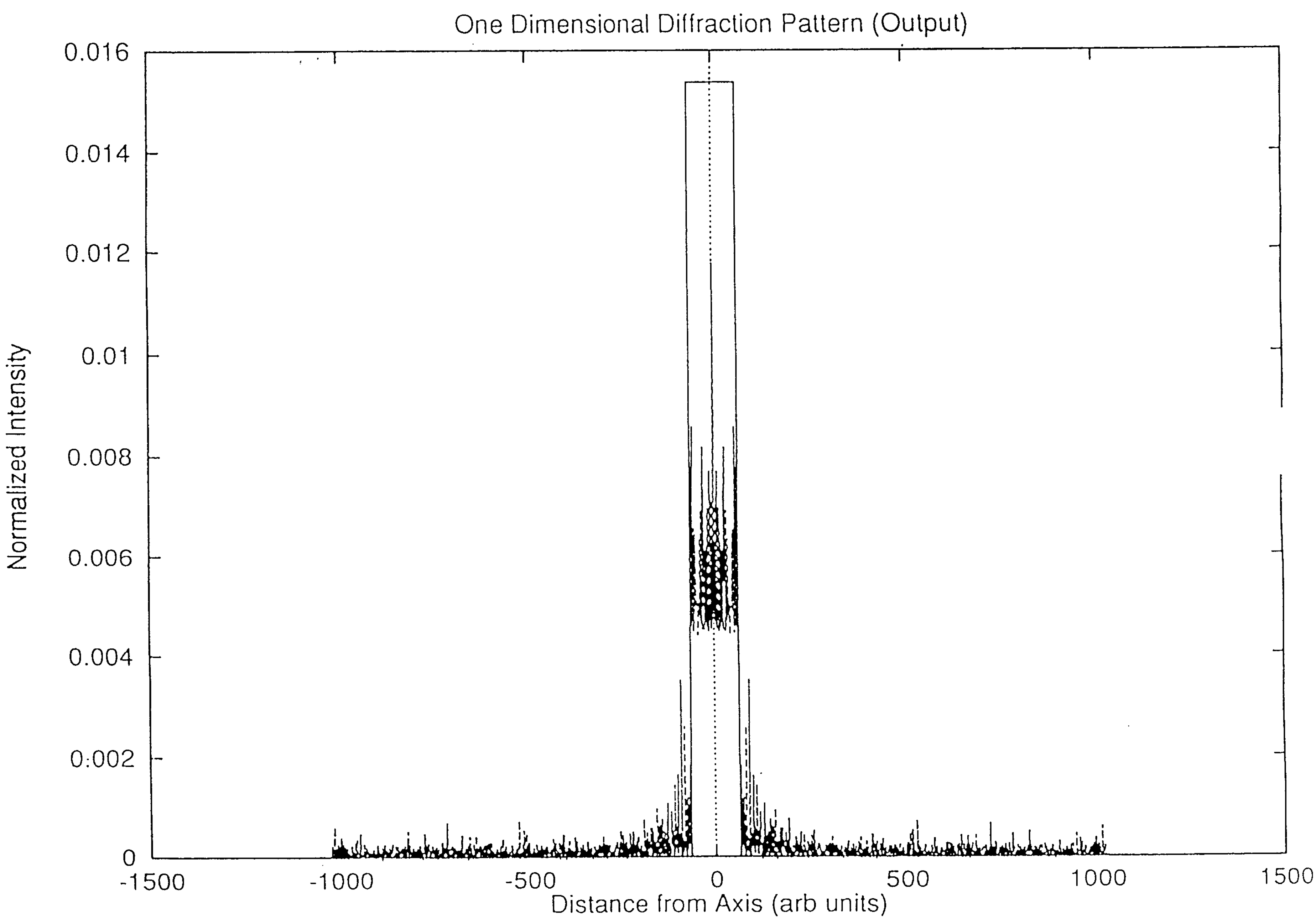


Fig. 4.100 65x65 One Dimensional Diffraction Pattern

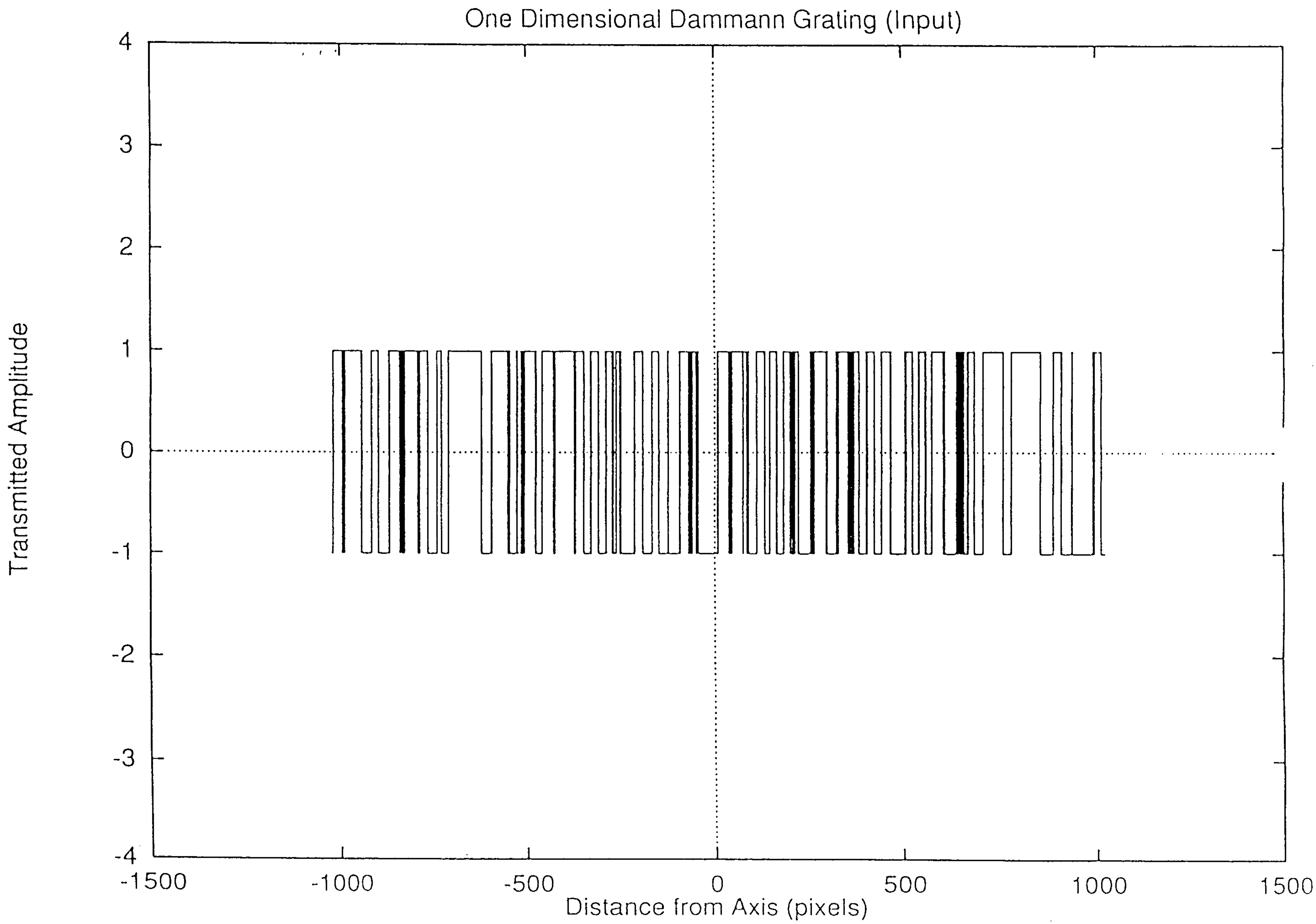


Fig. 4.101 65x65 One Dimensional Dammann Grating Structure

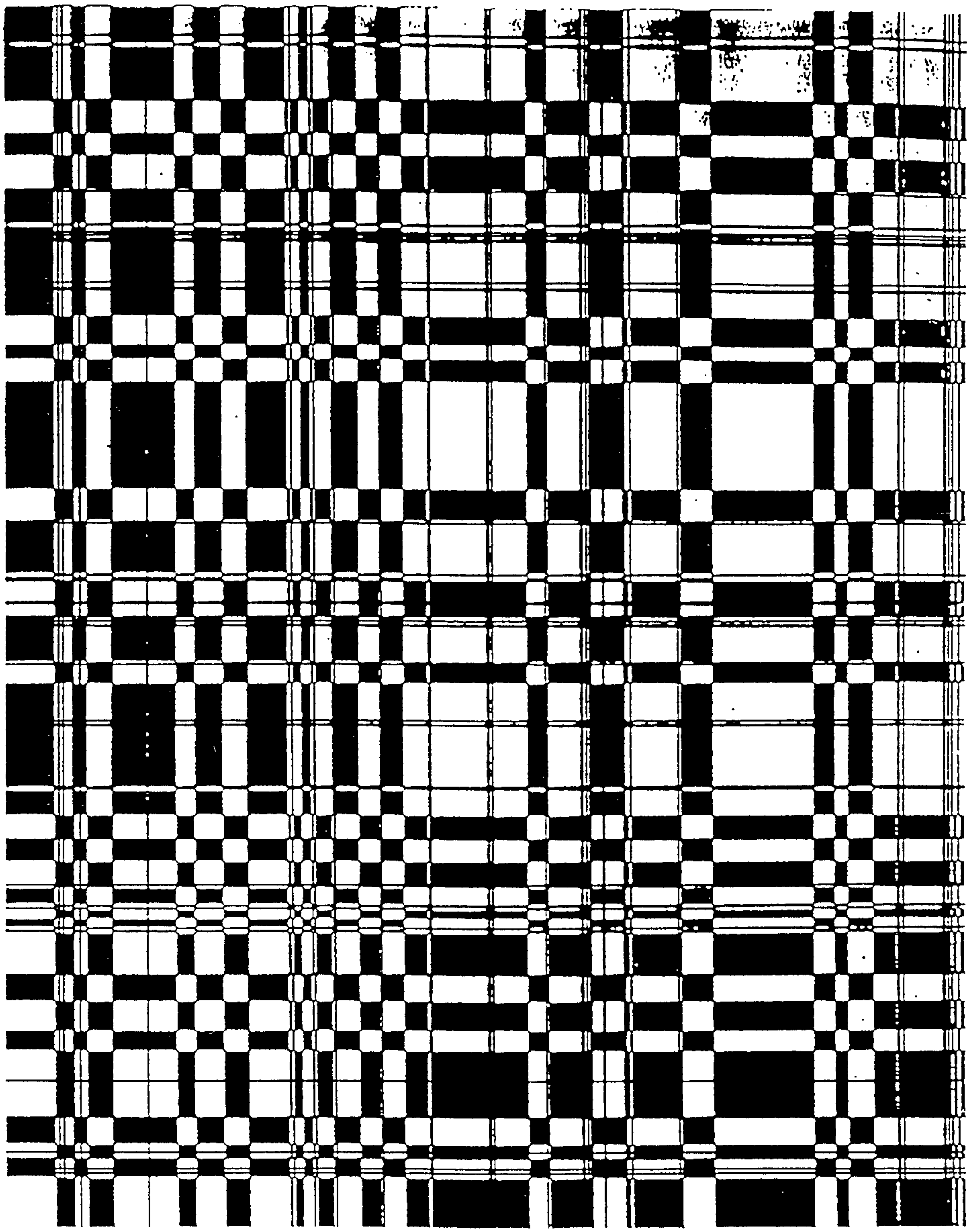


Fig. 4.102 65x65 Dammann Grating Cell

00000001 10111111 11111111 11111111 11111001 10011001 11111111
11111111 11111111 11110111 11111001 00010000 00000000 00000000
01111111 11111111 11111110 00000000 00001111 11111111 11111101
00000000 00000000 00111111 11111111 11101000 00100000 10111111
11000000 00000000 00000000 00000000 01110111 10000111 11111111
11111111 11111111 11110110 00000000 00000000 00000001 00110011
11111111 11111111 11111101 00100000 00101001 11111111 11100000
00000000 00000010 01111111 11111111 11100000 00000000 00000111
11111111 11111111 11110100 00000000 00000000 00000000 00000000
00100011 11111111 11111000 00000000 00000101 11111111 11111101
10000000 00000001 11111111 11111111 11111111 11101010 00000000
00000000 00000000 00001110 11111011 01101100 10000000 10101111
11111111 11000000 00000000 00000000 11111111 11111111 11111111
11111111 11111111 11111111 11110000 00000000 00000000 01111111
11111111 11111111 11111111 11111111 11111111 11111111 11111111
11111111 11111000 00000000 00000000 00000000 00000001 11111111
11111111 11101000 00000000 00000000 00000001 11000000 00000000
00000000 00000000 00000000 00000000 00000001 11110111 11111111
11111000 00000000 00000111 11111111 11111111 11111011 11110011
11111111 11111111 11111111 11111111 11111111 11010000 00000000
00000000 00000001 11111111 11111111 10000000 00000000 00000000
00000110 11111111 11111111 11111111 11001111 01101101 01111111
11111111 11111111 11111111 11011101 01111111 11111111 11111100
00000000 00000000 00000011 11111111 11000000 00000000 00000111
11111111 11111111 11111111 11111111 11111111 11111111 11111111
11111111 11111111 11111111 11111000 00000000 00000000 00000001
10111111 11111111 11111111 11111111 11111111 11100111 01000000
00000000 00101000 00000010 11011010 11111111 11111111 11111111
11110010 00000000 00000011 11111111 11111111 11111111 11110110
10111111 11111111 11111111 11111111 11111111 11111111 11101001
11111111 11111111 11110000 00000000 00000001 01111111 11111111
11100000 00000000 00000010 01111111 11111110 01100000 00011111
11001110 00000000 00000000 00000000 00000000 01011111 11111111
11111111 00000000 00000000 00000000 01111111 11111111 11000000
00000000 00000000 00010000 00000000 00000000 00000000 01011111
11111111 11111111 10000001 00001011 11111011 10111000 00000000
00000000 00000000 00000000 00000000 7.0906e-05 3364 299997

Fig. 4.103 65x65 Objective Function Value

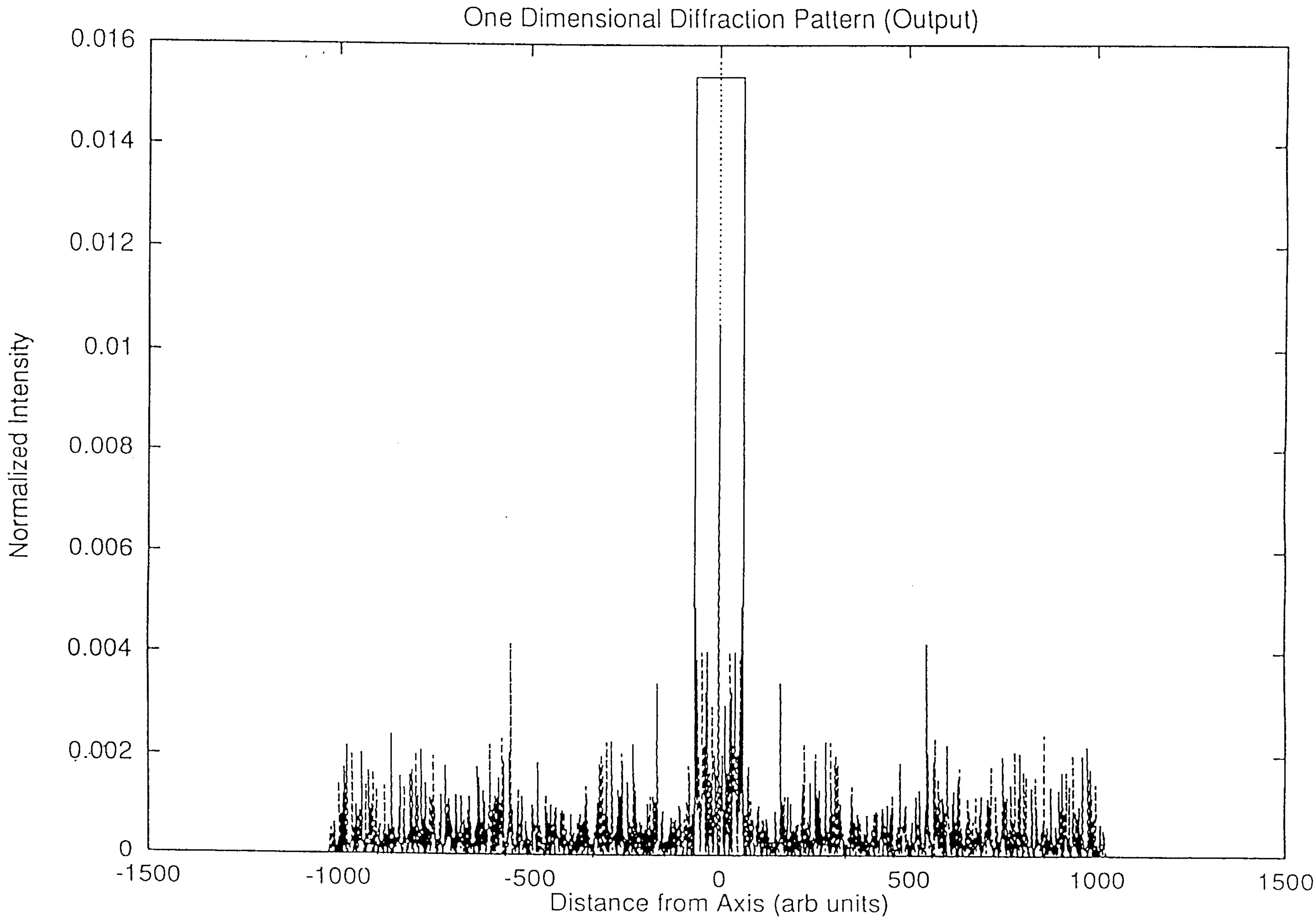


Fig. 4.104 65x65 One Dimensional Diffraction Pattern (High Mutation Rate)

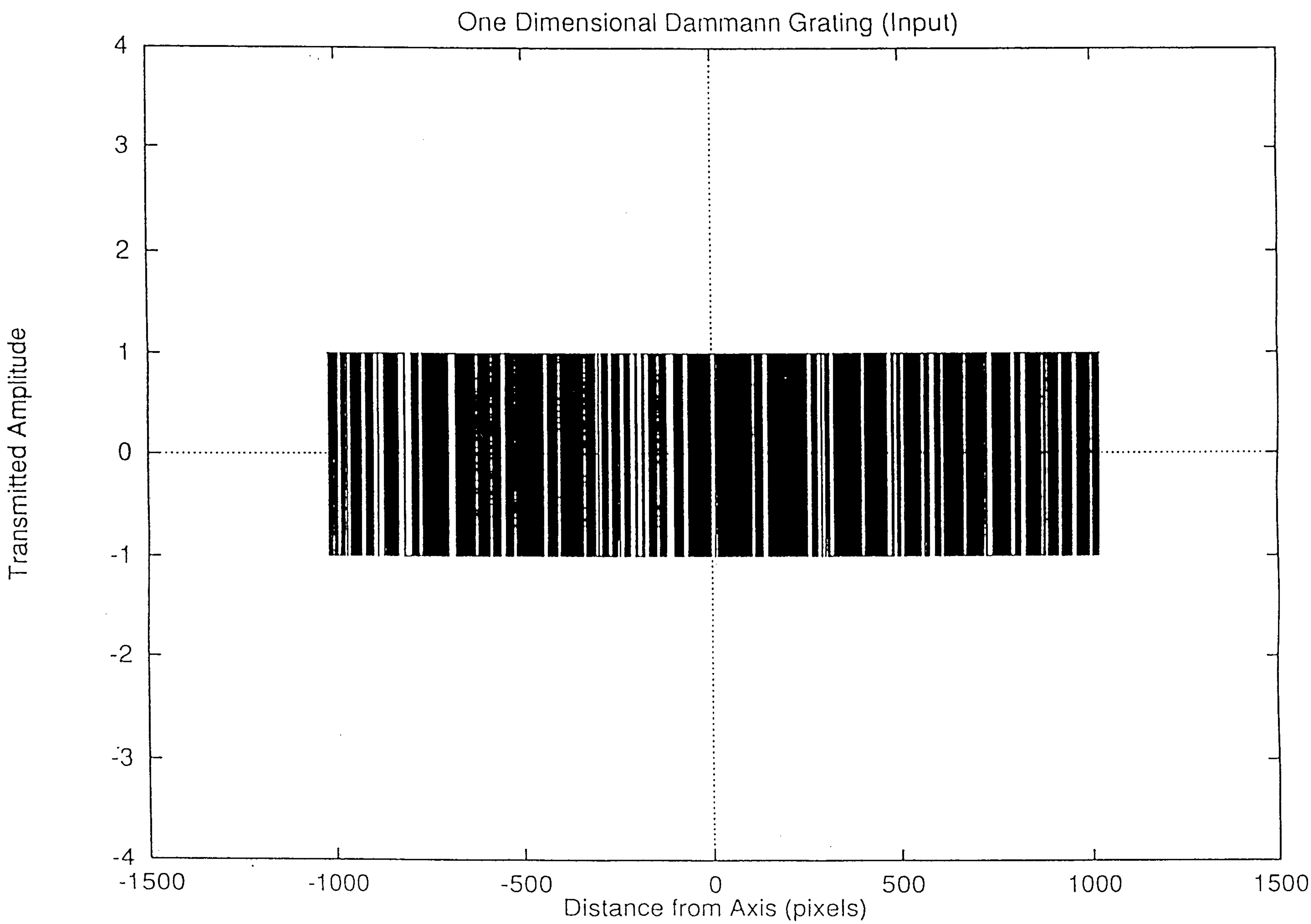


Fig. 4.105 65x65 One Dimensional Dammann Grating Structure (High Mutation Rate)

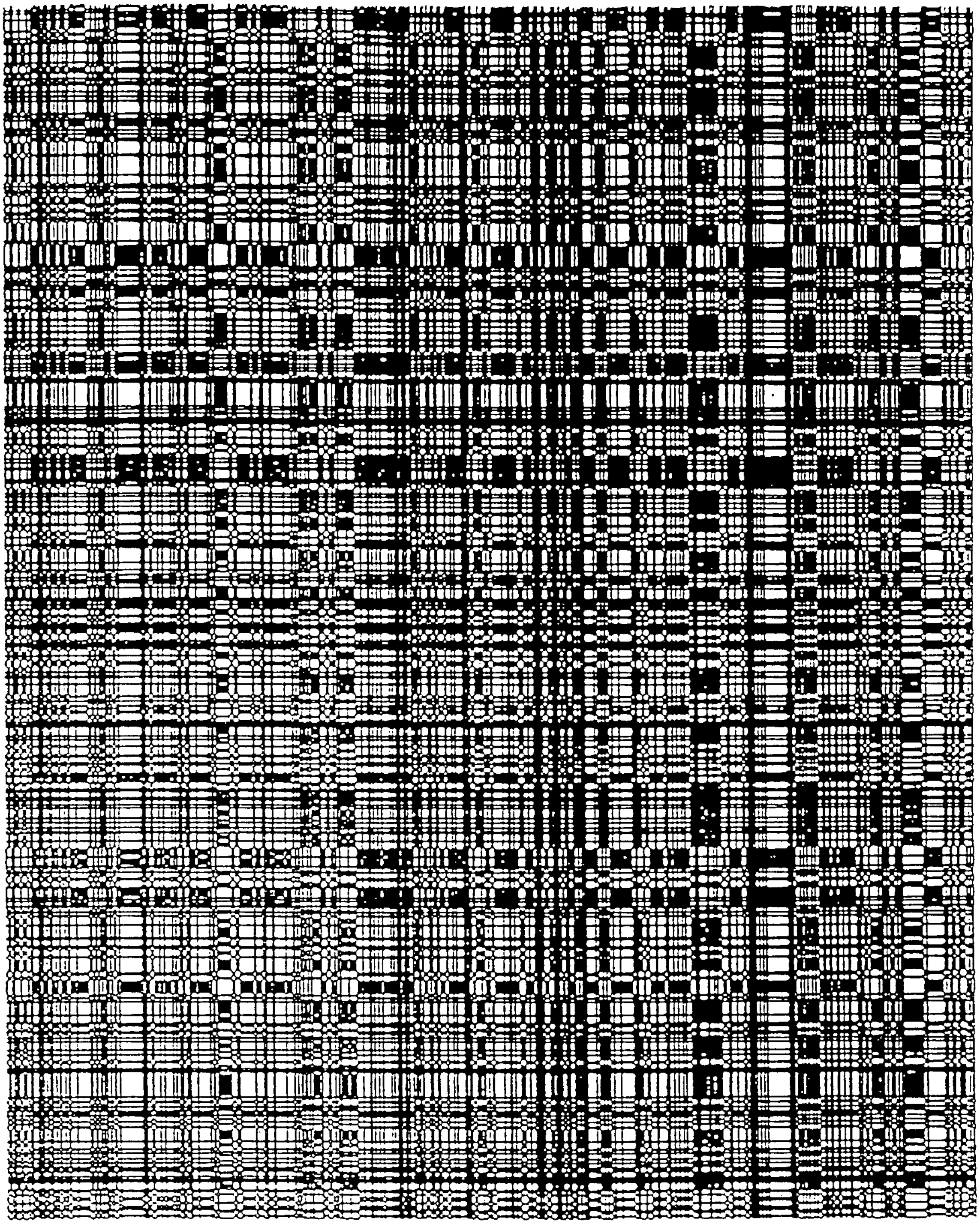


Fig. 4.106 65x65 Dammann Grating Cell (High Mutation Rate)

11111110 11111110 11110010 11010000 10001101 01111000 11011000
01011000 11101011 11110100 01100100 11000101 11110101 11101111
11011110 10111010 11100010 00111111 11111101 11011001 11001000
10001001 11101010 00010111 01100110 00101111 10110011 00110101
10110110 10000100 10110001 01111101 01111111 11101100 10001110
10000111 11111011 01111110 11111110 01110000 00000011 00100001
01001111 10101101 11011011 01110010 11000100 10101101 00011101
10110010 00000001 01000001 10110101 00100101 01100100 10101000
10011000 01000100 00100000 00000100 01010111 11101011 11001011
11111010 11110001 10110010 01101100 10101110 10111111 00111111
00000001 00110001 11111111 11000010 01101110 00011111 10001110
01010011 01100001 00101000 01111100 01101100 01110000 00011001
01001000 00100000 10010111 10101110 00111100 11111101 10000000
00000011 01101101 10001001 01100111 00001000 00111101 01010001
00000000 01000110 00111000 00111111 11111011 01000100 01010011
00001110 11101110 01000001 00000011 11011111 11011101 10010100
01111000 10000011 11111101 01110010 00001100 10010000 01011111
11111101 10100111 00010010 11100010 01111100 11001111 10100111
11011000 10010111 11110110 10000001 00000010 01000010 00000001
10000110 01011111 11011111 11011100 00101110 11001011 11011101
11011111 11110011 00100000 11011011 00001111 11111100 11111111
11111110 11111100 10011011 10000010 11000101 10010011 11111111
11111011 00000000 00000000 01011011 00001101 10100100 00000110
00110100 11111010 11101110 10111010 11010111 00010010 00100100
10001101 01011111 11111111 11111011 01101101 01010110 00111110
11110100 11110000 10010001 01000000 10101000 11110110 11101110
11101001 11111010 11110000 11000010 10011111 11111101 11110110
00100100 01101111 11101010 01001011 10100110 01010100 00111100
10101001 11101110 00001111 01101011 11011111 11100011 01100010
00001110 01010101 11111011 01111001 11110010 11001001 11100101
00010111 01010110 10101101 10111111 10111011 10110101 11110111
11000000 01000000 01001101 11000011 11010000 00010010 00100110
11001111 11011111 11110101 11101001 11110011 11111111 10000111
00000000 00001100 11011111 11111011 11101000 11010011 01101111
11011101 11100011 10101011 11111111 11111111 10110001 10110100
10011001 10001111 11111101 11110011 01110011 01101001 01101010
10111101 00110000 01100001 01100011 1.6025e-04 2879 287913

Fig. 4.107 65x65 Objective Function Value (High Mutation Rate)

Table 4.15 65x65 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(Two Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.70	"	6.8573×10^{-5}
100	0.70	"	7.5506×10^{-5}
200	0.70	"	7.8459×10^{-5}
50	0.50	"	6.6759×10^{-5}
100	0.50	"	6.9043×10^{-5}
200	0.50	"	7.4291×10^{-5}
50	0.60	"	6.8966×10^{-5}
100	0.60	"	7.9246×10^{-5}
200	0.60	"	6.7257×10^{-5}

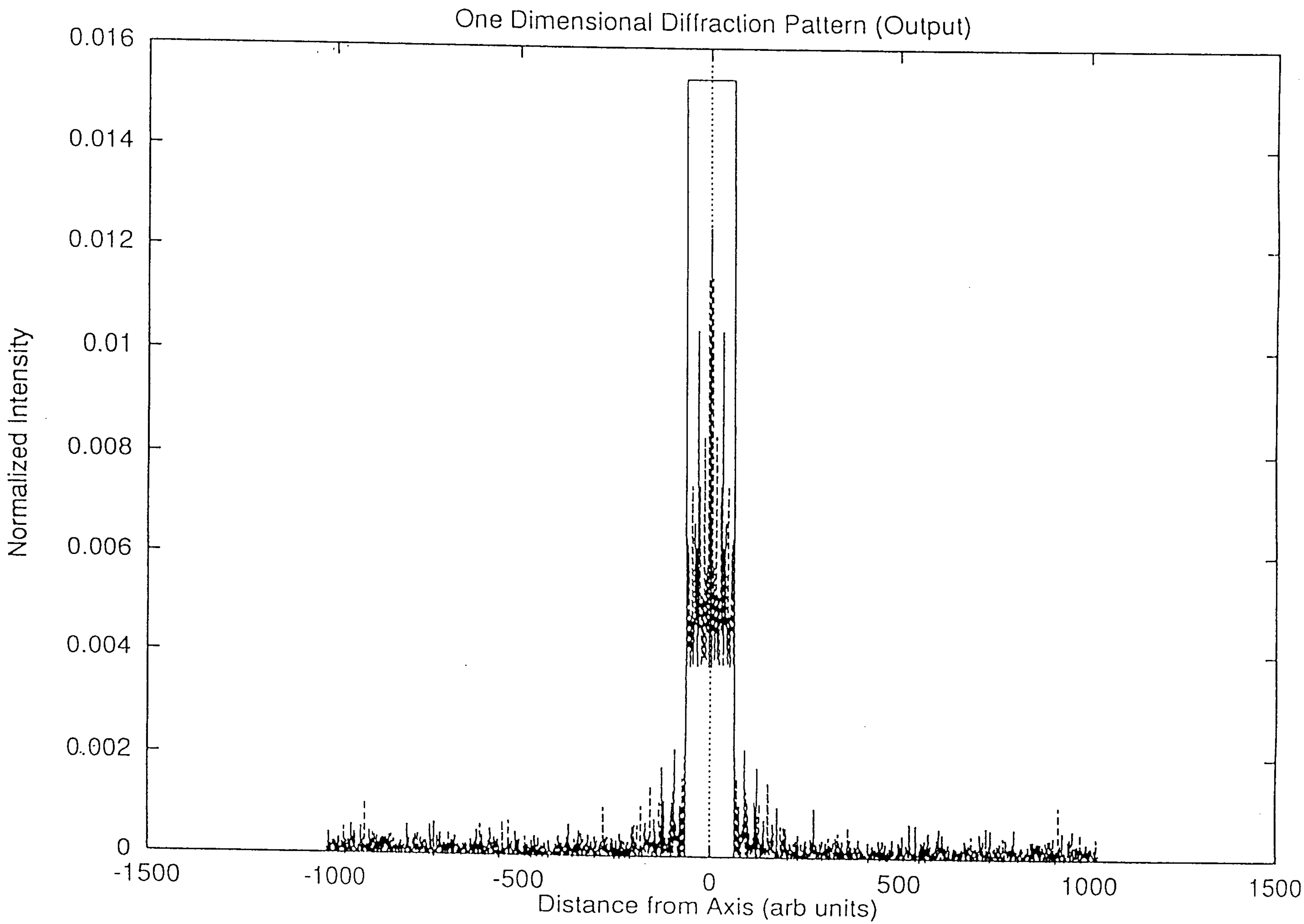


Fig. 4.108 65x65 One Dimensional Diffraction Pattern (Adaptive Mutation)

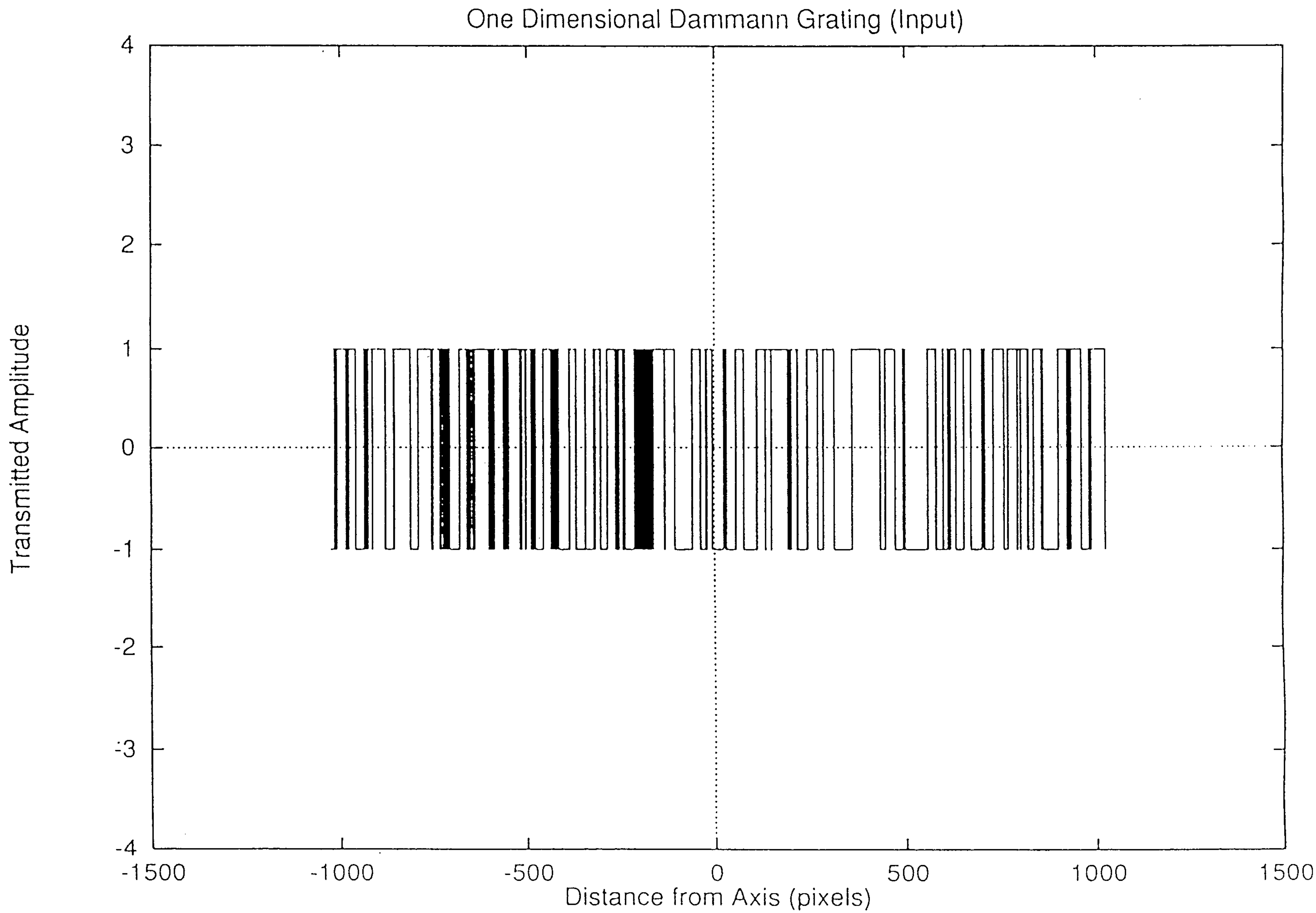


Fig. 4.109 65x65 One Dimensional Dammann Grating Structure (Adaptive Mutation)

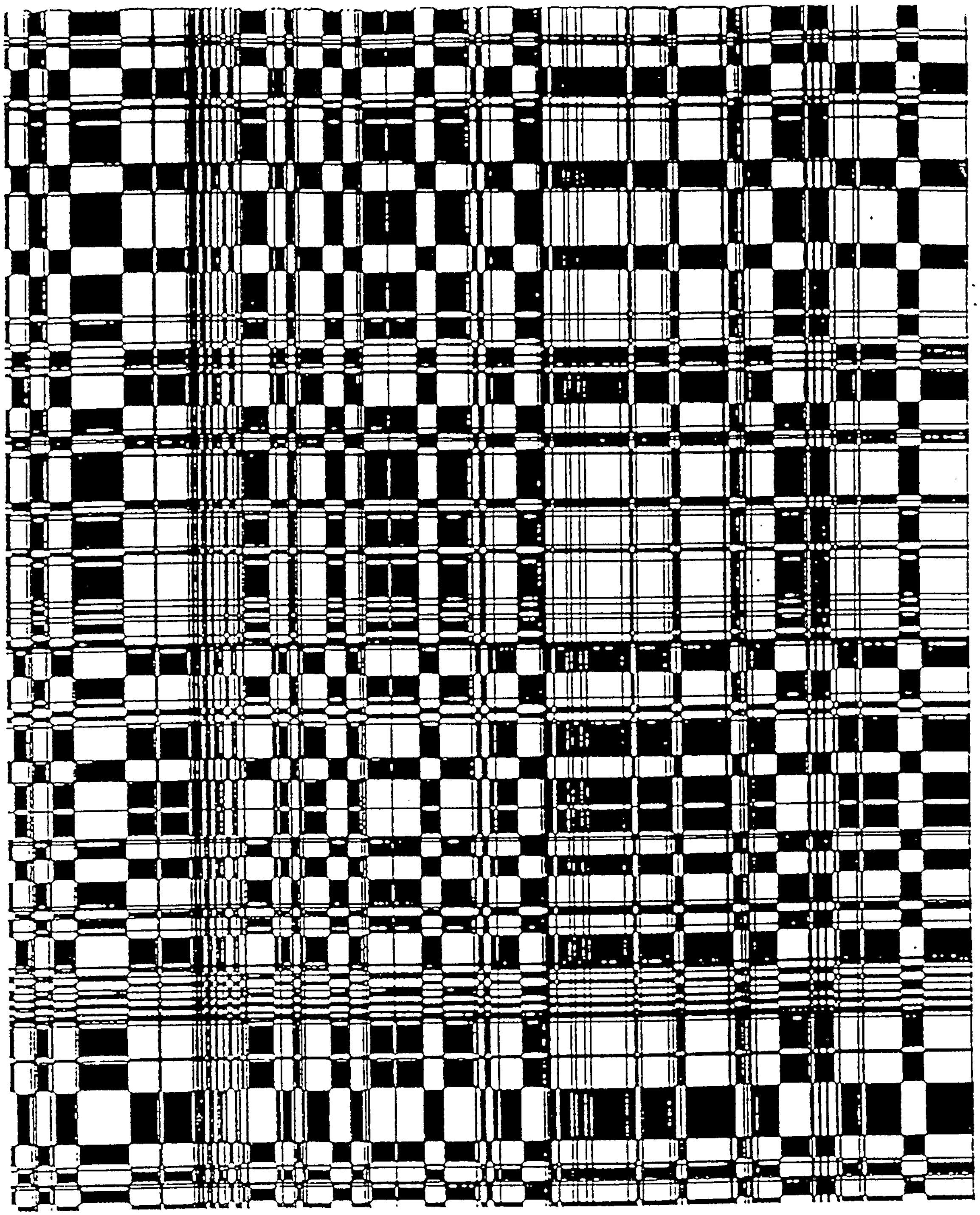


Fig. 4.110 65x65 Dammann Grating Cell (Adaptive Mutation)

00000000 00000000 00000001 11111011 00000000 00000000 00000010
01111111 11111111 11110000 00000000 00000000 00000000 00000110
11111111 11111111 11110111 11111111 11101111 11111111 11111111
11111111 11111111 11111011 00000111 01111111 11111101 01000000
00000000 00000000 11011111 11111111 11111111 11111000 00000000
00010111 11111111 11111111 11111111 10000000 00000000 00000000
00000000 00000000 00000000 11111111 11111111 11111111 11111111
11111111 11111111 11111111 11111111 11111111 10100000 00000010
11111111 11111111 11111111 00000000 00000000 00001110 11010000
00000000 00000000 00000000 00000000 00000000 00000000 00000101
11111111 11111111 11010100 00000000 00000001 00000000 00110010
11101111 11111111 00000000 00000000 00000111 11111111 11111111
00000000 00000000 00000000 00001011 11110100 00000000 00000000
00011111 11111111 11111111 11111110 00000000 10011111 11111111
11111111 10101111 11101111 11111111 11110110 10000000 00000111
11111111 11111111 10111100 00000000 00000000 00000000 00000000
00000101 11111111 11111111 11110001 01001111 01111111 11111111
11111111 11000000 00000000 00000110 11000011 11111111 11111111
11111111 11111111 00000000 01000000 10111111 11111111 11111111
00110010 01111111 11111111 11100000 00000000 00000000 01000011
11001011 11111111 01111111 11111111 11111111 11111111 10010000
00000000 00000000 11011111 11111111 11111111 11111111 11111111
11111000 00000000 00000000 00111111 11111111 11111111 11111111
11110011 10111111 11111111 11001110 00000101 00001010 00110100
00000000 00000000 00000000 11111111 11111111 11101110 10010010
00000111 01011111 11111111 11111111 11111111 11111110 10001000
11011101 11111111 11111111 11111101 10001011 11110111 11111111
11111111 11111111 11101111 01111111 01110111 11111111 00111110
10100100 00000000 00000000 00111111 11111111 11111101 11111000
00101111 00010000 00000000 00000000 00000001 01111111 11111111
10100000 00000000 00000000 00100000 00000000 00000000 01101101
11111111 11010000 00000000 00000111 11111111 11111111 11011010
00001011 11111110 11111000 00000000 00000000 00000101 00001101
11100000 11101111 01001101 01001101 01010110 11111111 11111111
11111111 11100101 11111111 11111111 11111111 00000000 00000000
00000000 00000000 00000000 00000101 11111111 11111111 10001000
00000000 10110111 11111111 11000000 7.9246e-05 4562 291799

Fig. 4.111 65x65 Objective Function Value (Adaptive Mutation)

Table 4.16 65x65 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Mutation Rate	Objective Function Value
50	0.70	0.001	6.6047×10^{-5}
100	0.70	0.001	6.6836×10^{-5}
200	0.70	0.001	7.1834×10^{-5}
50	0.50	0.10	1.5773×10^{-4}
100	0.50	0.10	1.6100×10^{-4}
200	0.50	0.10	1.6118×10^{-4}
50	0.60	0.001	6.7147×10^{-5}
100	0.60	0.001	6.6882×10^{-5}
200	0.60	0.001	7.0096×10^{-5}

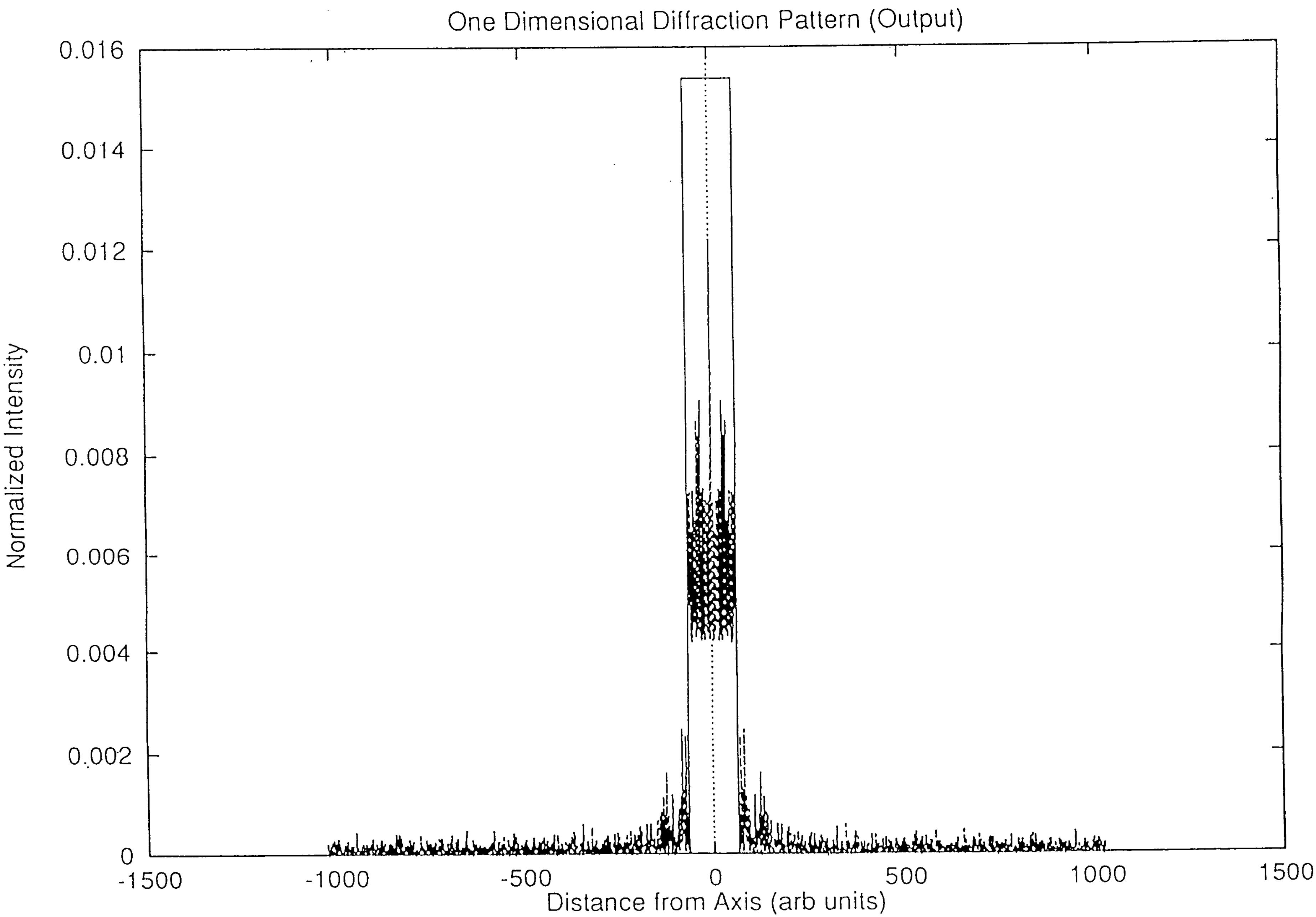


Fig. 4.112 65x65 One Dimensional Diffraction Pattern

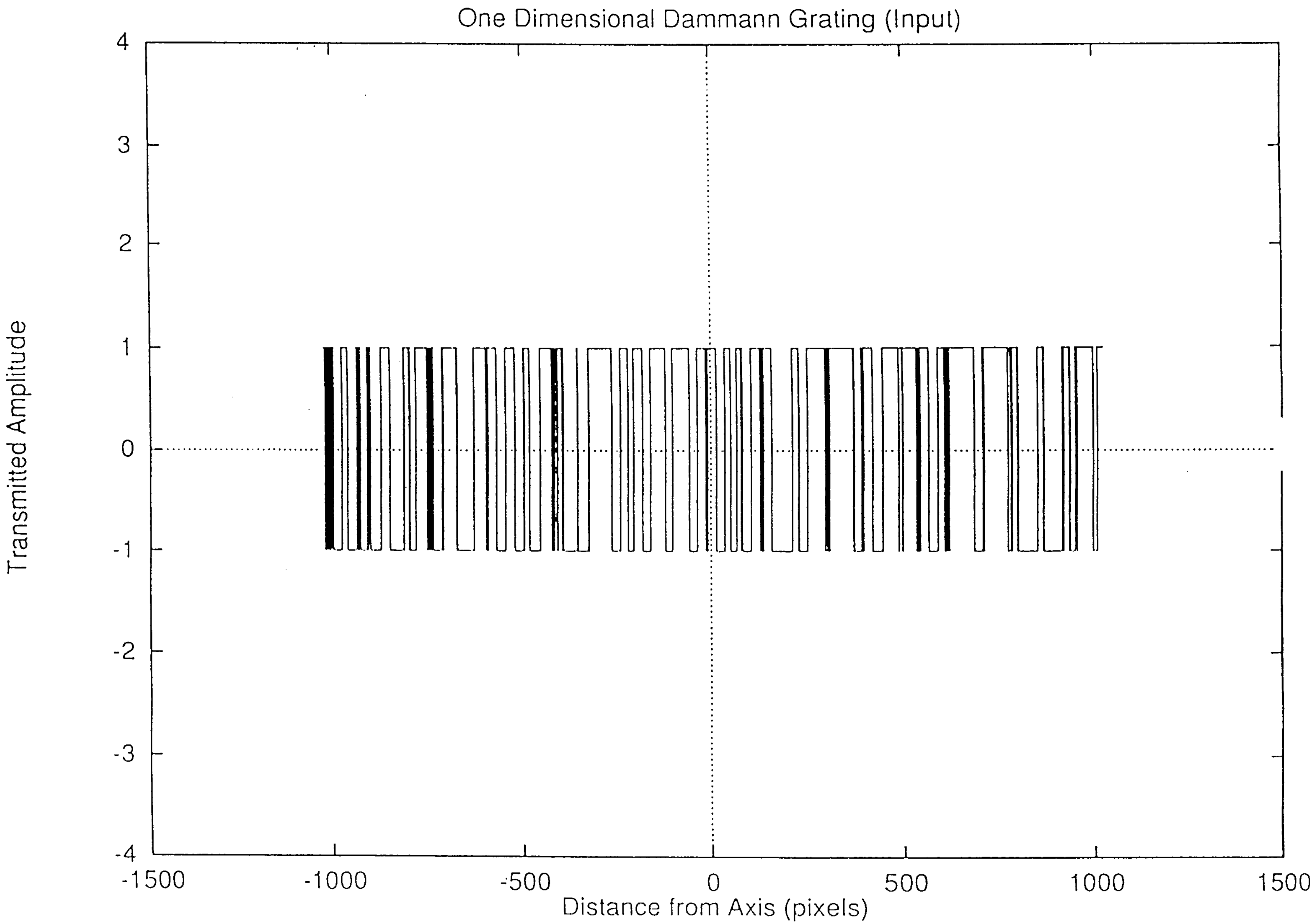


Fig. 4.113 65x65 One Dimensional Dammann Grating Structure

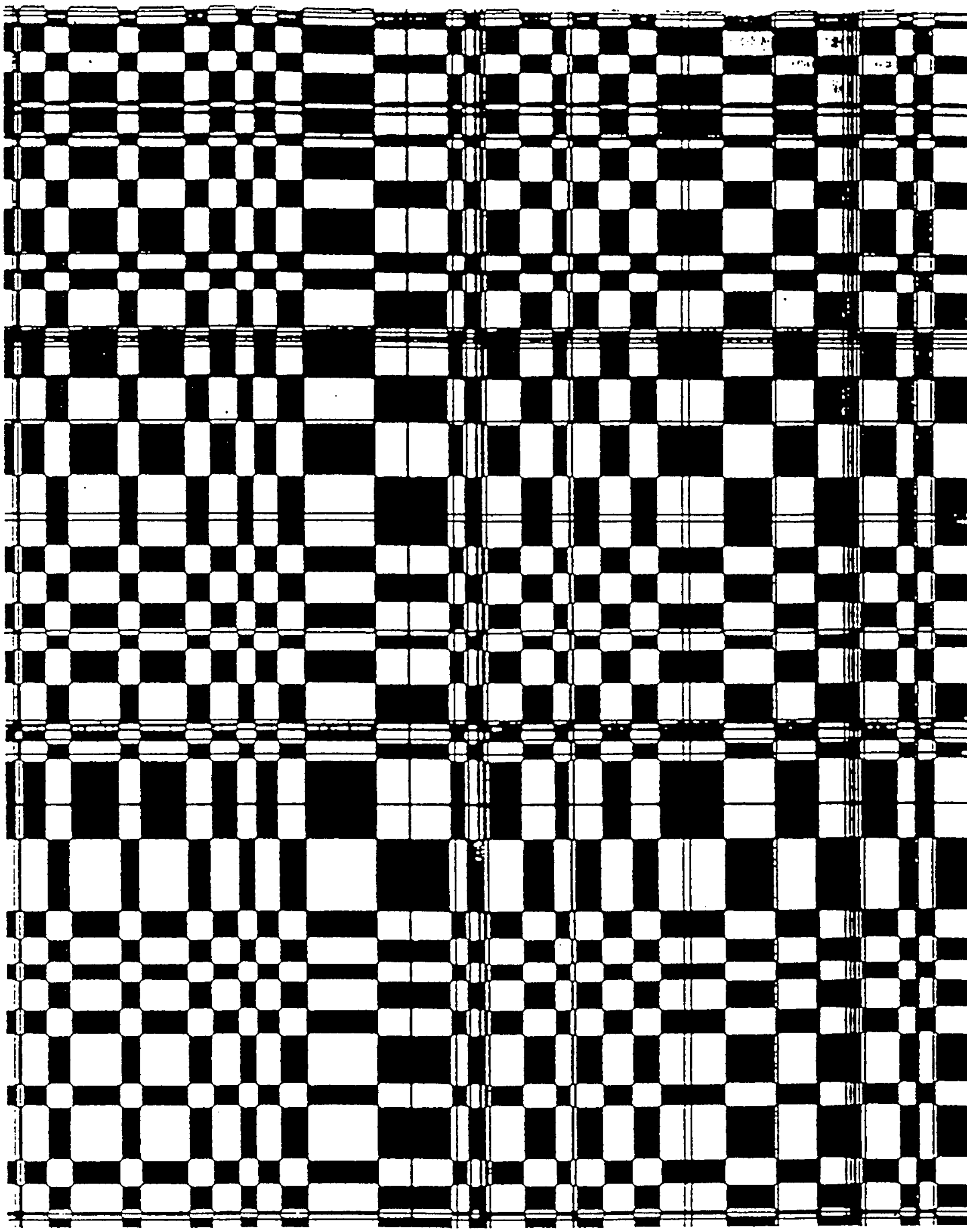


Fig. 4.114 65x65 Dammann Grating Cell

11111111 11111100 00000000 00000000 00000111 11111111 11110000
00000000 00000111 11111110 11111000 00000000 00000000 00011111
11111111 11111111 11011000 01000111 11111111 11111111 10000000
00000000 00000000 00000000 00000000 00000000 00000001 11111111
11111110 10000000 00000000 00000011 11111111 11111111 11111111
11111111 11111111 11100000 01100111 01111111 11111111 11111111
11111111 11111111 11111111 11111111 11110010 00000000 00000000
11011110 11111111 11111111 11111000 00000000 00000000 00000010
11111111 11111111 11111111 11111111 11111110 11111111 00111111
11111111 11111111 11111111 11100100 00010101 11111111 11111111
11100000 00000000 00000000 00111111 11111111 11011010 00010001
01111111 11111111 11111111 11111111 11111111 11111111 11111111
11111110 10000000 00000000 00000010 00001111 11111111 11111111
11111111 11111111 11111111 11111111 11111111 10000000 01001001
11111111 10111100 00000000 00000000 00000000 00000000 00000000
00000000 01111111 11111110 11000000 00000000 00000000 00000000
00000000 00000000 01101010 11111111 11110011 10000000 00000111
11011111 11111111 11111111 11111111 11111111 11110010 00000000
11111111 11111111 11101110 01000110 11101111 10101000 00000000
00000000 00111111 11111111 11000000 00000000 00000000 00101100
00101000 00000000 00001101 01111110 00000000 00000000 00000000
00111111 11111111 11111111 10000000 00000000 00000000 00000000
00000110 11111111 11101010 00000000 00000111 11111111 11111111
11111111 11011001 00101001 00010000 00000000 00000000 00101011
11111111 11111111 11111111 11111110 11000000 00000000 00000000
00000000 00000000 00000111 11111111 11111111 11111111 11110111
11011111 11111111 11111111 00000000 00000000 00000001 11111111
11111111 11111111 00000000 00000000 00000011 00111111 11111110
00000000 00000000 00000000 00011111 11111111 11111111 11111111
01101001 01000000 10011111 11110111 10000000 00000000 00000000
00000000 00001010 00000000 00000000 00000000 00101111 11111111
11111111 11111111 11111111 11111111 11111111 11111110 10000000
00000000 00000001 11111111 11111111 11100000 00000000 00111111
11111111 11111111 11000000 00000000 00000001 11111111 11111111
11111111 11111111 11111111 00000000 00000000 00111111 11111111
11111111 11111111 11111111 11111110 00000000 00000000 00001111
11111111 11111111 10010011 01111111 7.1834e-05 1675 298933

Fig. 4.115 65x65 Objective Function Value

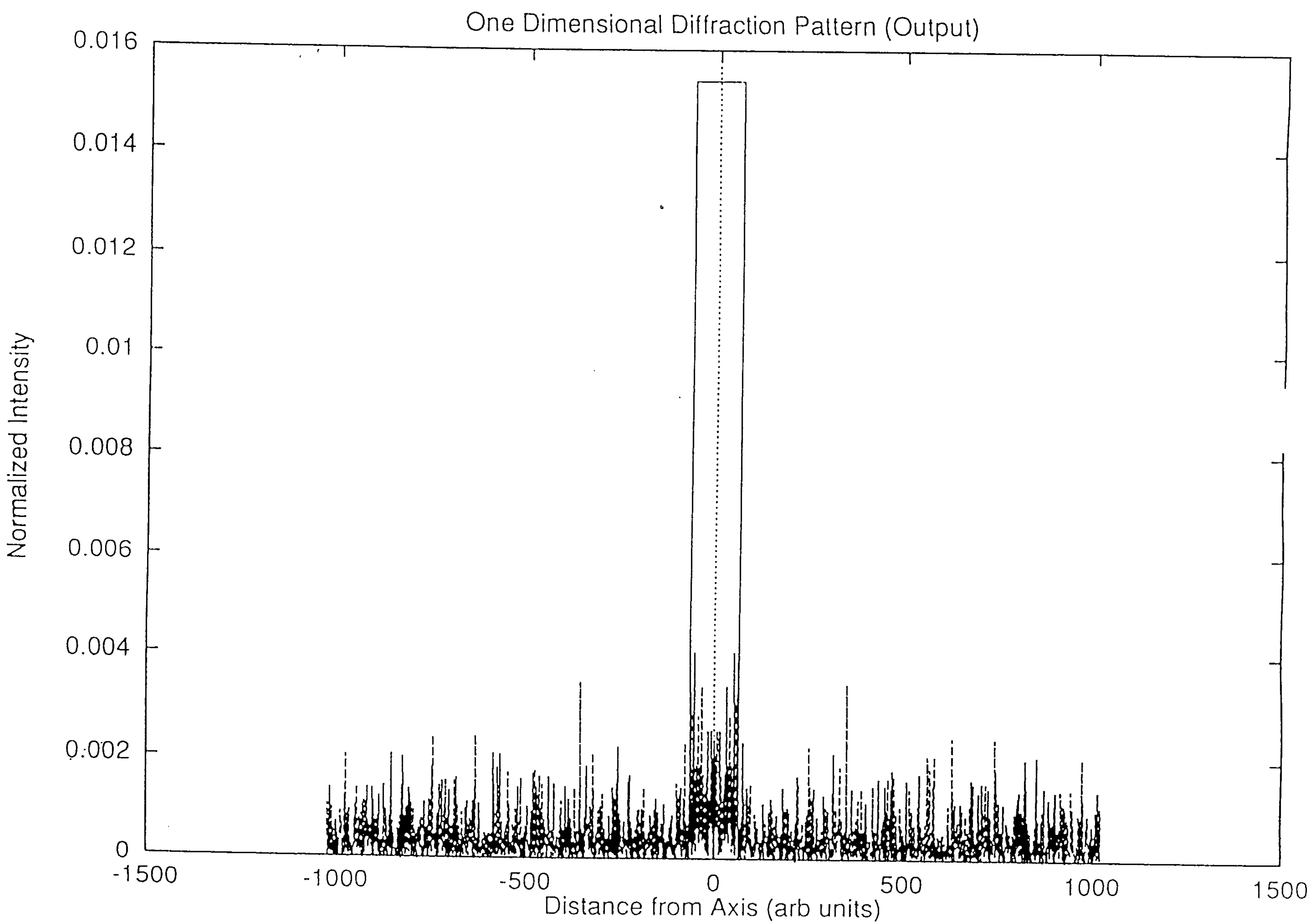


Fig. 4.116 65x65 One Dimensional Diffraction Pattern (High Mutation Rate)

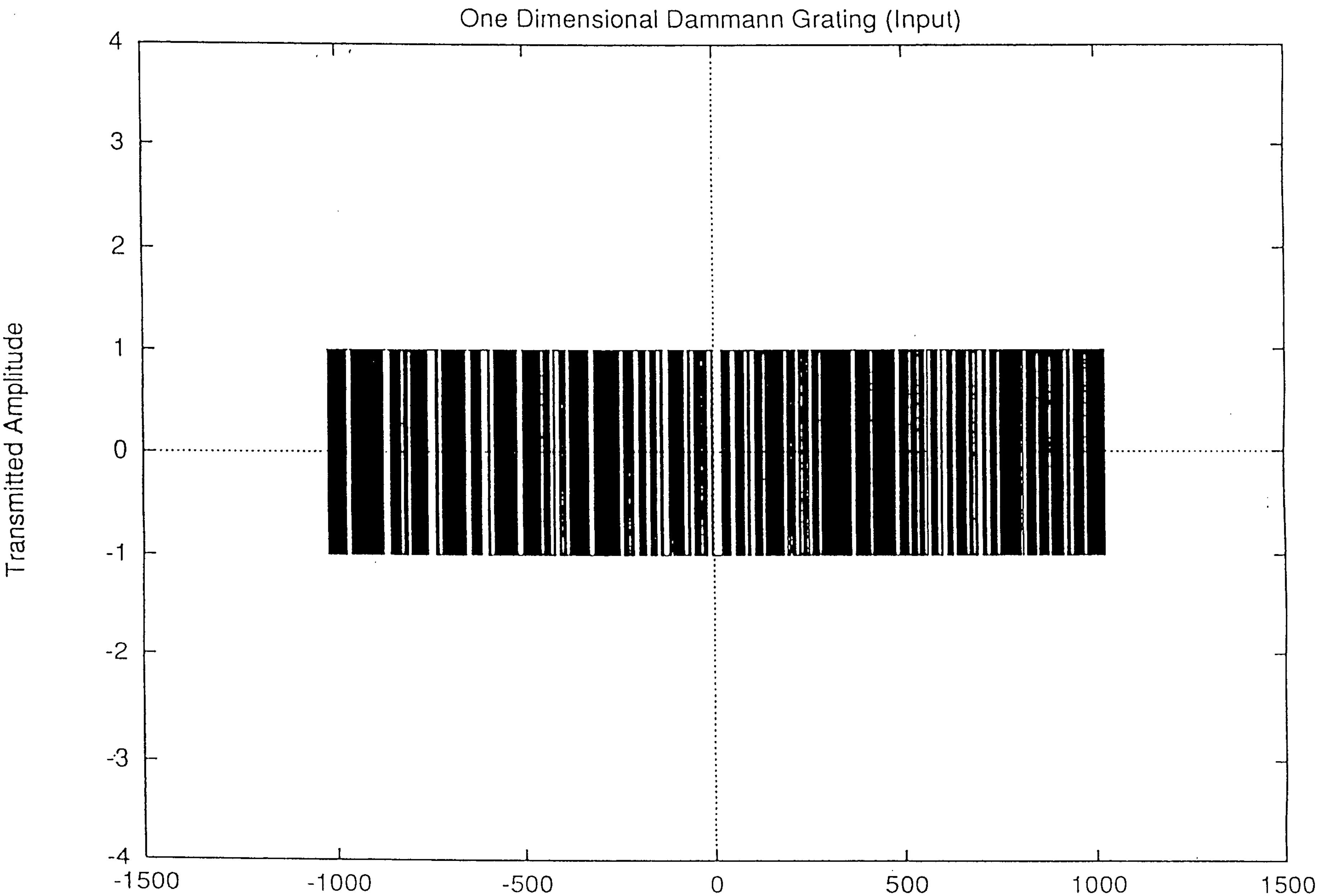


Fig. 4.117 65x65 One Dimensional Dammann Grating Structure (High Mutation Rate)

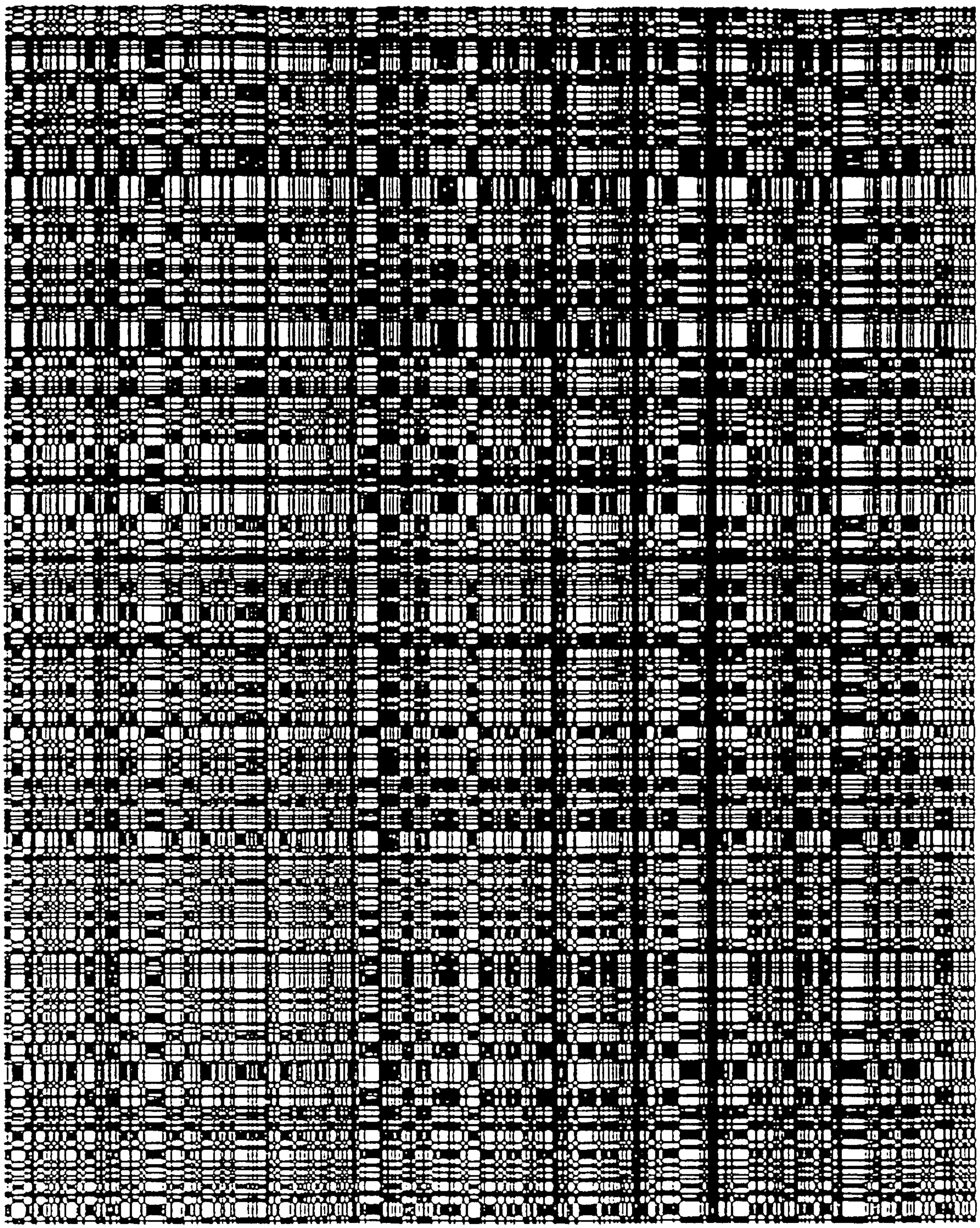


Fig. 4.118 65x65 Dammann Grating Cell (High Mutation Rate)

00000000 00000000 00001000 10100111 01011110 10110111 11111111
10111101 01101101 01100111 10011111 11011111 10111111 11110111
00001101 10110100 11011111 10100010 01010011 10001011 11000110
10110010 10100110 01011111 11011111 10011111 10011100 10000000
01010010 00001101 01011100 00001000 00001010 01101101 00001101
11110111 10010010 11001010 10111011 10111011 01110101 11101110
11101101 00100110 01100101 11010111 11111101 10110000 10001001
01011011 11001010 11001011 11101011 10101111 00111001 11011101
00101000 11000111 10100110 00111010 00000000 11001011 01001100
11110011 00000011 00111100 00100100 00001000 01011010 11000000
11111111 10111101 11010011 00110000 00010110 11111111 11001101
10110101 10011111 10000100 01001110 00111001 01001100 00001110
01010100 00001011 10011111 11111101 01101000 11000000 01011010
00001111 00101000 00100101 00000111 10111000 10011100 01000100
01000110 10011100 01111111 01111111 00101001 00001110 11000101
00011111 01011111 01110100 00111000 00101111 11010101 01100001
00101110 01110101 11011111 11101111 10111000 00001101 10001001
10011011 11000010 01000000 10010001 10001111 10101111 01101110
11110111 10010110 10001000 00101000 10000110 10011100 01010101
10110111 01011111 11111100 10110101 10100001 00000100 01100001
11010010 11010111 00010110 11101001 00001000 00100001 01001011
11111111 11111111 01111100 11011000 11100101 00010100 00000111
10011100 01111110 11011011 11010010 01111011 01111101 01001001
10011011 11111111 11111111 11010101 11010000 00101001 11100001
00010010 01010111 10111010 01101100 00101100 00010100 00011111
11111111 00111010 11110101 01010110 01101111 11111111 11111000
00100000 00001100 01011111 00101001 01010100 11100110 11001001
00010000 10011100 01000000 00000001 11100001 01101011 01010100
00001010 01001101 00110111 00000001 00100011 10000110 10000000
01001011 11111111 00011100 00001001 01000000 01000001 11011010
11011000 10100100 01101110 11011010 11010100 00000000 00100001
10101011 10011001 10110000 01001101 10011111 00011001 10111011
10001101 11111111 00001100 01010111 11110111 01110111 11111111
10010011 11001001 11100111 11111101 00110001 00000011 01111111
10111100 10000000 00000000 11000111 11101011 11110100 10001010
01101011 01000000 00111000 10111111 11001110 10011100 11011000
00011010 10000011 11111111 10001000 1.6118e-04 1128 225773

Fig. 4.119 65x65 Objective Function Value (High Mutation Rate)

Table 4.17 65x65 Dammann Grating Arrays with different Genetic Algorithms settings (GenesYs 1.0)

(One Point Crossover)

Population Size	Crossover Rate	Adaptive Mutation	Objective Function Value
50	0.70	"	6.6047×10^{-5}
100	0.70	"	7.2367×10^{-5}
200	0.70	"	7.3438×10^{-5}
50	0.50	"	7.1614×10^{-5}
100	0.50	"	6.7920×10^{-5}
200	0.50	"	7.4520×10^{-5}
50	0.60	"	7.1242×10^{-5}
100	0.60	"	7.6626×10^{-5}
200	0.60	"	7.2326×10^{-5}

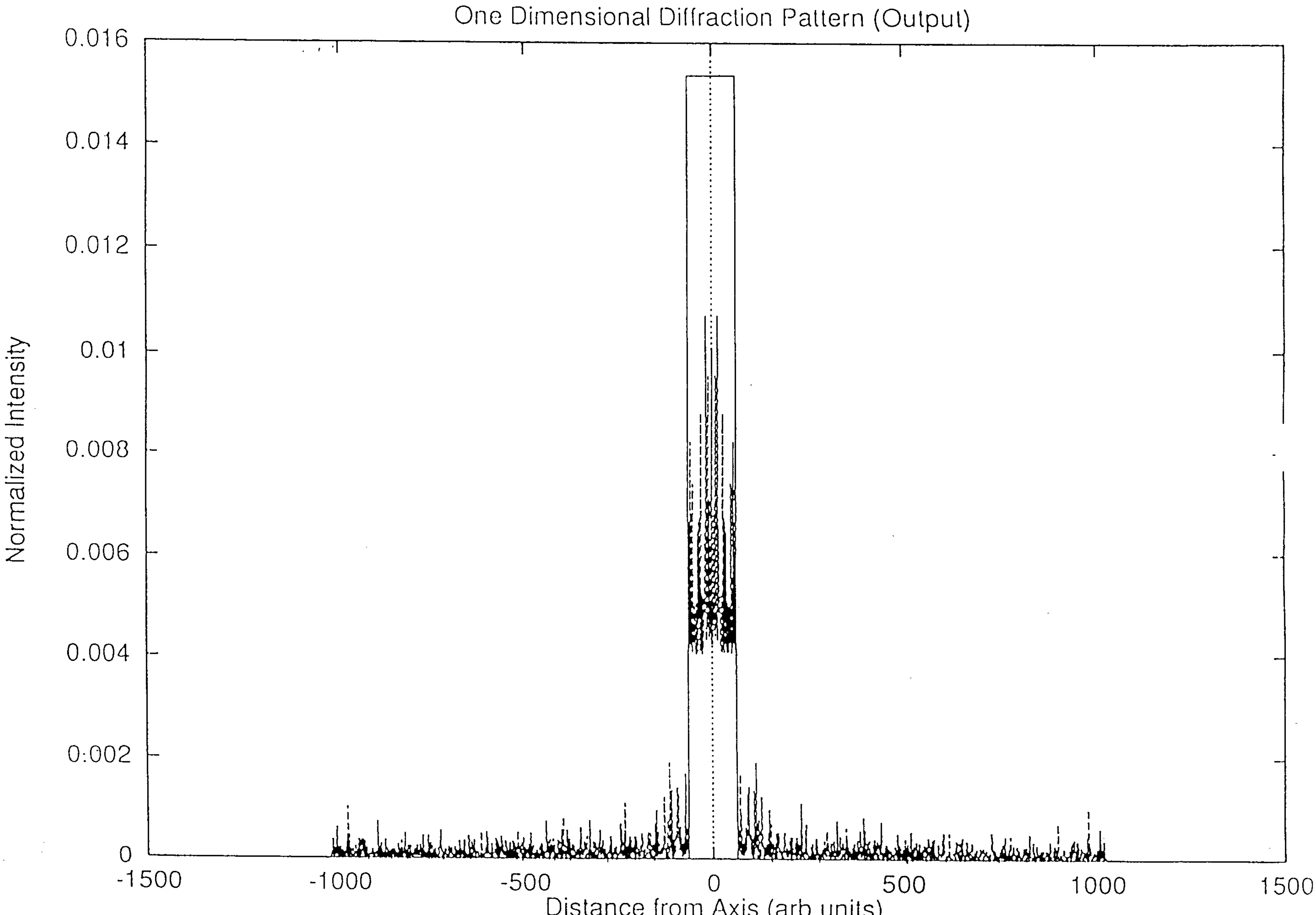


Fig. 4.120 65x65 One Dimensional Diffraction Pattern (Adaptive Mutation)

4.98, and 4.99 respectively. The diffraction efficiency is estimated to be about 50% and fair resolved diffraction pattern. The input Dammann Grating structure of figure 4.97 is also fairly resolved.

4.4 65 x 65 Amplitude Dammann Gratings

The generation of this amplitude gratings requiring a bit string of 2048 was also carried out using the GenesYs 1.0 package. The schemes used were two point crossover, one point crossover, standard and adaptive mutation mechanisms. The simulation was also done on a Sun workstation with each run carried out with 300,000 trials. Time of computation increased as well as the degree of computation. For a population size of 100, 70 per cent crossover rate, and 0.1 per cent mutation rate, the results obtained are as shown in table 4.14 and in figures 4.100, 4.101, 4.102, and 4.103 respectively. The resolution of the diffraction pattern as observed in figure 4.100, showed a very low resolution. The diffraction efficiency is estimated to be about 35%. The input grating of figure 4.101 is of very low resolution. The low resolution could be attributed to the size of the grating and perhaps the number of trials used in the simulation. With higher number of trials, it is envisaged that the resolution will drastically improve and hence produce a better Dammann Grating. Shown in figures 4.104, 4.105, 4.106, and 4.107 are the results from high mutation rate. As anticipated, the resolution of the diffraction pattern is very low and diffraction effificiency about 7%. Figure 4.105 which shows the input grating, it is evident from the figure that the resolution is quite low and that the grating structure would produce less resolved grating if it is fabricated as amplitude or phase grating Simulating this devise with adaptive mutation method and two point crossover, the results obtained are shown in table 4.15 and in figures 4.108, 4.109, 4.110, and 4.111 respectively. The resolution and diffraction efficiency resulting in this case are quite similar to that obtained with the two point crossover and low standard mutation rate. Also, this grating was generated using one point crossover for standard and adaptive mutation mechanisms. The results obtained for the point crossover standard mutation are shown in table 4.16 and in figures 4.112, 4,113, 4.114 and 4.115 respectively. Again, the resolution and diffraction efficiency are similar to that obtained with two point crossover low standard mutation rate. The diffraction effficiency is estimated to be about 45%. Shown in figures 4.116, 4.117, 4.118 and 4.119 are the results obtained with high mutation rate. Finally, shown in table 4.17 and figures 4.120, 4.121, 4.122, 4.123, and 4.124 respectively. Figure 4.123 is the fabricated 65x65 amplitude grating produced by Imagesetting Graphics Workshop of Syracuse, New York. Again, the resolution of the diffraction pattern as observed in figure 4.120 is very low and the diffraction efficiency is about 40%.

All the simulations for 9x9, 17x17, 33x33 and 65x65 size Dammann Gratings were done for 300,000 trials on the Sun workstation. The results of one point crossover mechanism, low standard and adaptive mutations were not significantly different from the results of the two point crossover, low standard and adaptive mutations. As could be observed from the figures showing results from the high standard mutation rate, high mutation rates generate less efficient and noisy device. Therefore, the use of high mutation rates in Genetic Algorithms for the generation of this optical device is not recommended for further research in this field. The use of very low mutation rates or the use of adaptive mutation mechanisms are highly recommended for efficient system. Population size plays critical role in the application of Genetic Algorithms for optimization purposes. For my particular case, it was observed that using population of size 100 for most of the simulations, gave the best result. But, in some of the simulations, population size of 50 or 200 produced a good result. This is attributed probably to the mutation mechanism employed. High crossover rate in all cases produced good results. Very low crossover rate, however, was not experimented with in this project. For further research in this field, it will be interesting to explore the results that could be obtained with low crossover rate. The very low resolution obtained in especially the large size grating such as the 33x33 and 65x65 could be attributed to low number trials. Higher number of trials such as one million trials for the large size grating, is envisaged to produce high resolution grating with high diffraction efficiencies.

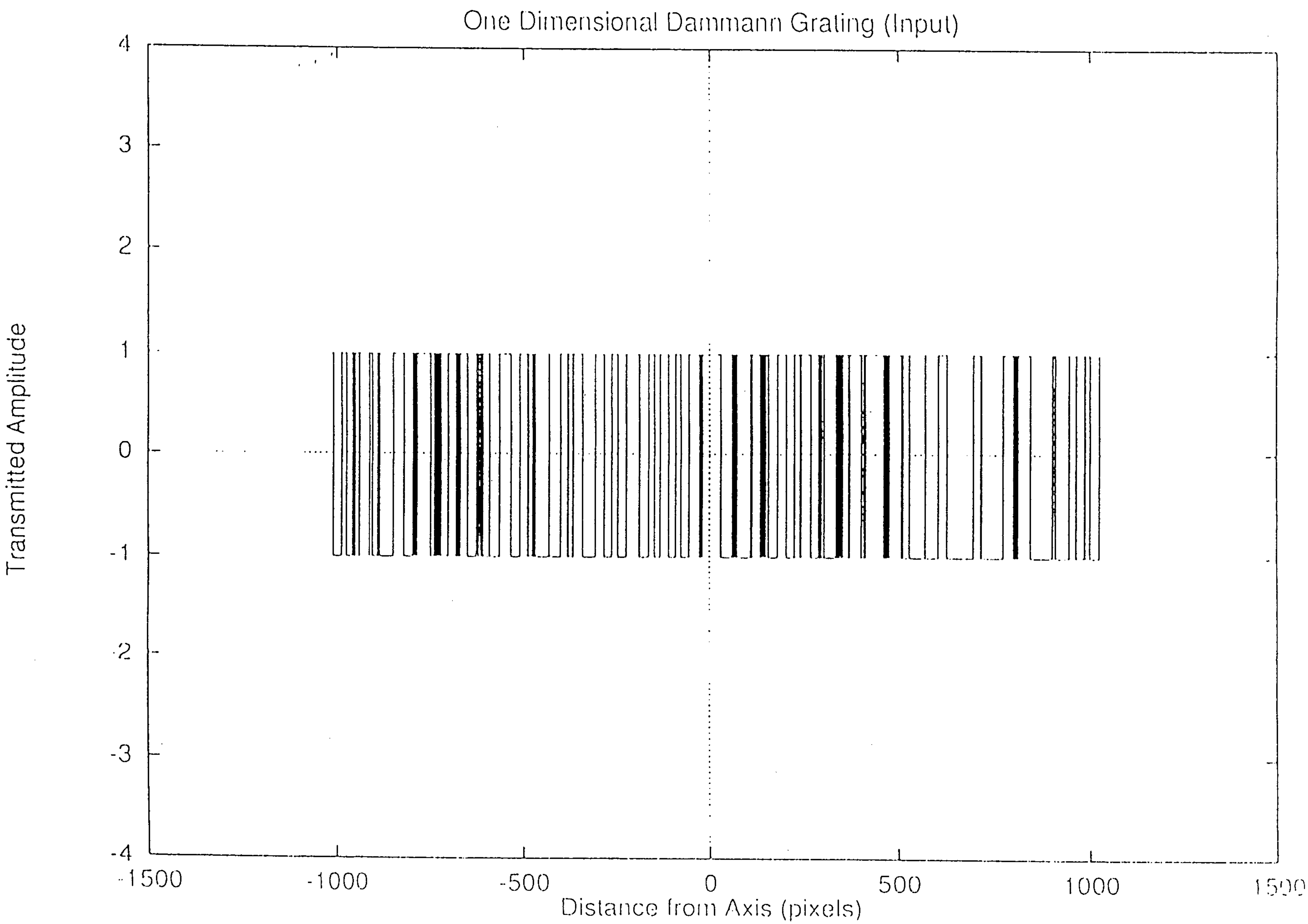


Fig. 4.121 65x65 One Dimensional Dammann Grating Structure (Adaptive Mutation)

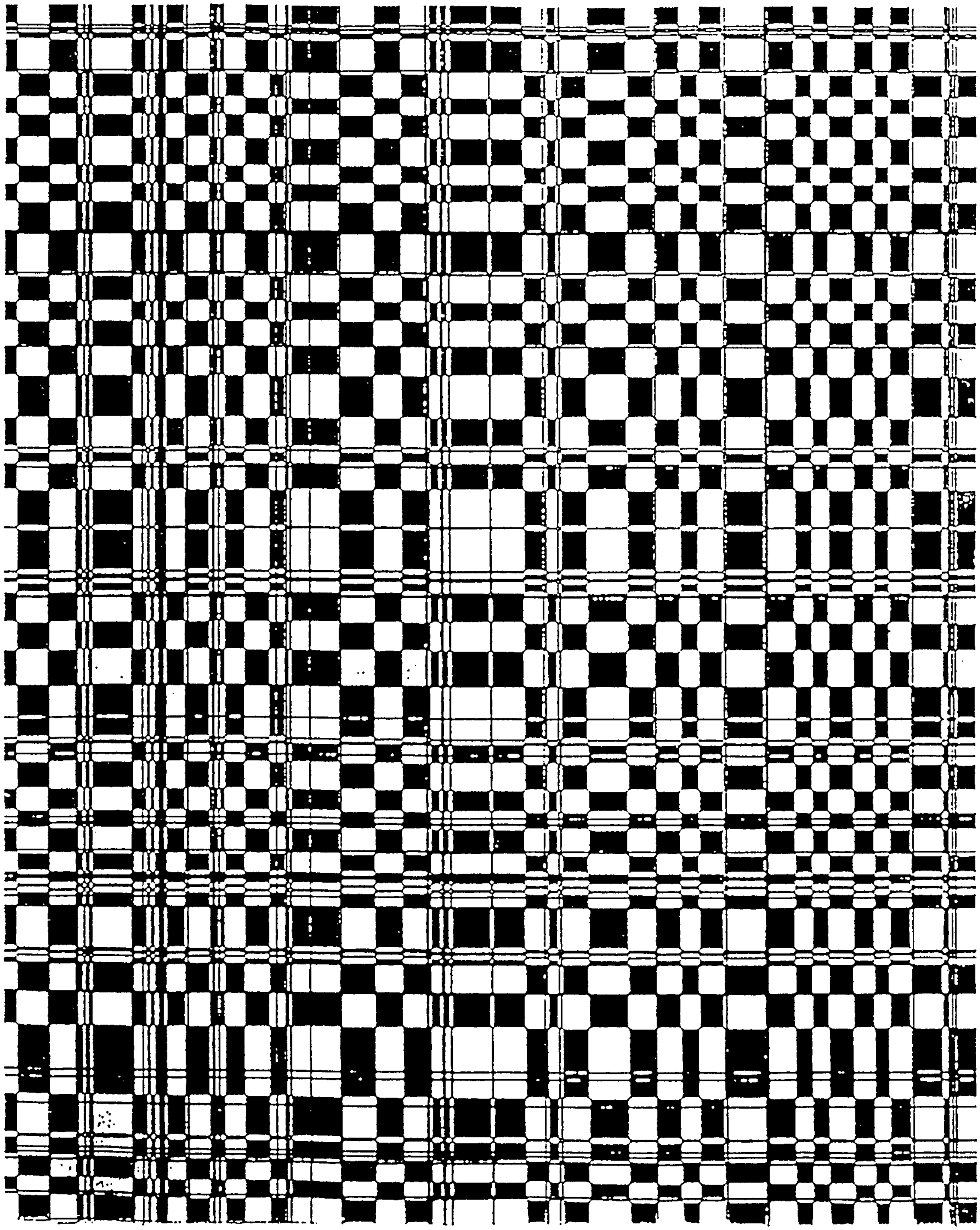
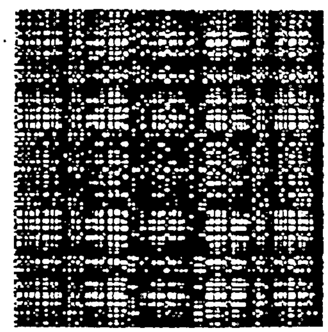


Fig. 4.122 65x65 Dammann Grating Cell (Adaptive Mutation)



114.97 4096 doveelec.ps Dove Electronics 336-0230

Fig. 4.123 65x65 Fabricated Amplitude Dammann Grating
(Adaptive Mutation)

11111111 11111111 11111111 11111001 00000000 00000000 00000000
00011011 10001100 01100000 00000000 00000000 00000000 00001010
00001111 11111111 11111110 11100011 11011011 00000010 00000000
00000000 00000001 11111111 11111111 11111100 00000000 00000000
00110111 11111111 10001001 11111111 11111111 11111110 10000000
00000000 00011100 11011000 00010000 00000000 00000000 00000000
01000011 10010000 01110101 11111111 11101110 11111111 11111111
11111111 11101100 00001010 10111111 11111111 11111111 11111111
11111111 11111011 10100011 01011001 11111111 11111111 11111111
11101010 10000000 00000000 10000000 00000000 00000000 00000000
00000000 01000000 00000000 00000000 00000000 00011111 11111111
11111111 11000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000111 11111111 11111100 11010000 00000000
00000000 00000000 00000000 00000000 00000000 01111111 11111111
11111111 11010110 00010101 00011111 11111111 11111111 11111111
01100000 00000000 00000000 00000000 00000000 00000000 00000000
00111011 11100110 00000000 00000000 00000000 00000010 01100000
00000000 01000100 00000000 00000000 11110111 11111110 10000000
00000000 00000000 11111111 11111110 01000000 00000000 00000011
01111111 11111000 00000000 00001001 11011101 11111110 00111111
11111111 11111111 11110111 11110000 00000000 00100000 10000000
00000000 00000000 00000000 00000111 11111111 11111111 11111111
01000000 00000000 00000000 10111100 11010011 11111111 11111111
11111111 11111110 10000000 10001110 00011101 01011000 00000000
00001101 11111111 11111111 10010100 01000001 11111111 11111111
10000000 00000000 00000001 11111011 11111010 01100000 00000000
00010000 00000000 00000000 00000101 11111111 11111111 11111111
11110000 00000000 00000000 00011111 11111111 11111110 10110000
00001101 00001100 00000000 00000000 00000000 00000000 11000000
00000000 00000000 00000000 11111111 11111111 11011100 00000001
10011111 11111111 11111111 11100000 00000000 00000000 00000000
00000011 11111111 11111111 11111010 00000000 00000000 00011111
11111111 11100000 00000000 00000000 11011111 11111111 11111111
11111111 11010100 00000000 00000000 00000111 11111111 11110000
00000000 00011111 11111111 11111111 11000000 00000000 00000111
11111111 11100000 00000000 00000001 10111111 11111111 11111111
11100001 01101101 11111111 11111111 7.6626e-05 3952 252270

Fig. 4.124 65x65 Objective Function Value (Adaptive Mutation)

CHAPTER FIVE

CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

The goal of this work was to explore the possibility of using Genetic Algorithms to design and fabricate a binary optical element for application in optical interconnections. The binary optical element designed and fabricated in this research effort is the so-called "Dammann Grating" after the German holographer, Hans Dammann, who first proposed the optical device in the early 1970's. This device as was noted earlier, generates an array of equal intensity light spots from a single incident beam. Also, the use of Spatial Light Modulators for on line evolution of optical interconnections and the proper location of the optical device on an optical processor was studied. The algorithm used to design the binary device "Genetic Algorithm" which is based on the principles of natural selection and the survival of the fittest, was developed also in the 1970's by John Holland and colleagues at the University of Michigan. In the SLM study, it was observed and concluded that, a potentially valuable architecture in the first instance, will employ a single SLM placed in the optical system's fourier plane or in between two lenses. It was also found that in this architecture, coherence requirements are neither necessary nor harmful for the interconnection to take place. Other architectures studied employed two SLMs for the optical processor. In one case, the first SLM is placed in the Fourier plane while the other is inserted in the image or conjugate plane. This geometry, it was observed, could convert one arbitrary 2-dimensional pattern into another. Here, coherence is absolutely required. In the final case, one of the SLMs is placed in the image and the other in the Fourier plane. Again, this architecture requires that the illuminating beams be mutually coherent. And the mask written on the SLMs should be phase-only mask.

The binary optical device designed with the Genetic Algorithm requires that an objective function be established. With this fact, therefore, a novel objective function was formulated by us and termed "Multiple Objective Functions". The objective functions were categorised into additive and multiplicative methods. For this work, only the additive method was explored. The multiplicative method is left for further experimentation using Genetic Algorithms.

Using Genetic Algorithm software package obtained from the public domain, I have established the possibility of using the algorithm to design an optical device. I used the algorithm to design a 9x9, 17x17, 33x33, and 65x65 Dammann Gratings. One of the packages used for simulation is the Genesis 5.0. This package was only used to design or simulate a 9x9 grating one point crossover scheme. The results obtained in this case, were quite impressive. The diffraction efficiency was about 90% and the diffraction pattern was highly resolved. Another package, GenesYs 1.0 was used. Here, this package was used to generate a 9x9, 17x17, 33x33 and 65x65 Dammann Gratings. This package was used because of its various features which were not included in Genesis 5.0 and the fact that I could not obtain resolvable patterns with large size arrays such 17x17 and so on.. The features, include m-point crossover , adaptive mutation, proportional selection, linear ranking, uniform ranking and inverse linear ranking to name a few.

In generating the above mentioned size array Dammann Gratings, features provided in the GenesYs 1.0 such as m-point crossover and adaptive mutations were used. One and two point crossover schemes were used only to determine whether there would be significant differences in results obtained with the two schemes. As was observed from the results, this was not the case. The results from both schemes did not differ significantly. Another interesting phenomena was also observed when low standard mutation and adaptive mutation were used. In all cases, whether one point crossover or two point crossover, the results remain largely the same. This points to the fact that, adaptive mutation only incorporates low standard mutation rate to the genotype at all times. And this indicates that, the algorithm at all times seeks low standard mutation rate for its processes in evaluating stated objective functions. As was observed in all the simulations applying high standard mutation rate, the algorithm produced noisy diffraction patterns and low diffraction efficiency. Therefore, it is concluded here, that the use of high standard mutation rate in evaluating objective functions in Genetic Algorithms is not recommended. Very low standard mutation rate such as 0.001 is highly recommended. Population size of 100 in most of the simulations carried out for this work gave the best result. Therefore, it is also concluded that this size of population is the best for evaluating objective functions in Genetic Algorithms. Population size above 100 was found in some cases to produce undesirable results. In the case of crossover rate, it was found that high crossover rate gives good objective function results. Low crossover rate in all cases in the simulations it was found, gave less objective function results

For further experimentation using Genetic Algorithms, it is suggested that using the additive method of the multiple objective functions, more simulations be carried out using more trials or number of generations than were used in this work and also to try the other values of alpha. For this work, I have used alpha equals to 0.5. It is also suggested that, the multiplicative method and the second approach of our multiple objective function be experimented on and the results compared with results I obtained in this work by using the additive method. Also, as a further work on this project, it is suggested that the generated gratings be optically evaluated, by using the optical setup shown in figure 3.1 in chapter three and the diffraction efficiencies compared with the computer generated results.

At this juncture therefore, I would like to state that, it has been shown and proven that, it is possible to use an optimization algorithm such as Genetic Algorithm to generate devices, and in this project, an optical device. The evaluation function algorithm developed for this work found in appendix A, is capable of generating any desired size Dammann Grating.

APPENDIX A

/* This program evaluates only the additive objective function proposed by us for the generation of the binary optical element for this research work. *\

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#define PI 3.141592653589793
#define SWAP(a,b) tempr=(a);(a)=(b);(b)=tempr

void four1(float data[], unsigned long nn, int isign)
{
        unsigned long n,mmax,m,j,istep,i;
        double wtemp,wr,wpr,wpi,wi,theta;
        float tempr,tempi;

        n=nn << 1;
        j=1;
        for (i=1;i<n;i+=2) {
                if (j > i) {
                        SWAP(data[j],data[i]);
                        SWAP(data[j+1],data[i+1]);
                }
                m=n >> 1;
                while (m >= 2 && j > m) {
                        j -= m;
                        m >>= 1;
                }
                j += m;
        }
        mmax=2;
        while (n > mmax) {
                istep=mmax << 1;
                theta=isign*(6.28318530717959/mmax);
                wtemp=sin(0.5*theta);
                wpr = -2.0*wtemp*wtemp;
                wpi=sin(theta);
                wr=1.0;
                wi=0.0;
                for (m=1;m<mmax;m+=2) {
                        for (i=m;i<=n;i+=istep) {
                                j=i+mmax;
                                tempr=wr*data[j]-wi*data[j+1];
                                tempi=wr*data[j+1]+wi*data[j];
                                data[j]=data[i]-tempr;
                                data[j+1]=data[i+1]-tempi;
                                data[i] += tempr;
```

```
                        data[i+1] += tempi;
                    }
                    wr=(wtemp=wr)*wpr-wi*wpi+wr;
                    wi=wi*wpr+wtemp*wpi+wi;
                }
                mmax=istep;
        }
}
#undef SWAP

void realft(float data[], unsigned long n, int isign)
{
        void four1(float data[], unsigned long nn, int isign);
        unsigned long i,i1,i2,i3,i4,np3;
        float c1=0.5,c2,h1r,h1i,h2r,h2i;
        double wr,wi,wpr,wpi,wtemp,theta;

        theta=3.141592653589793/(double) (n>>1);
        if (isign == 1) {
                c2 = -0.5;
                four1(data,n>>1,1);
        } else {
                c2=0.5;
                theta = -theta;
        }
        wtemp=sin(0.5*theta);
        wpr = -2.0*wtemp*wtemp;
        wpi=sin(theta);
        wr=1.0+wpr;
        wi=wpi;
        np3=n+3;
        for (i=2;i<=(n>>2);i++) {
                i4=1+(i3=np3-(i2=1+(i1=i+i-1)));
                h1r=c1*(data[i1]+data[i3]);
                h1i=c1*(data[i2]-data[i4]);
                h2r = -c2*(data[i2]+data[i4]);
                h2i=c2*(data[i1]-data[i3]);
                data[i1]=h1r+wr*h2r-wi*h2i;
                data[i2]=h1i+wr*h2i+wi*h2r;
                data[i3]=h1r-wr*h2r+wi*h2i;
                data[i4] = -h1i+wr*h2i+wi*h2r;
                wr=(wtemp=wr)*wpr-wi*wpi+wr;
                wi=wi*wpr+wtemp*wpi+wi;
        }
        if (isign == 1) {
                data[1] = (h1r=data[1])+data[2];
```

```
			data[2] = h1r-data[2];
		} else {
			data[1]=c1*((h1r=data[1])+data[2]);
			data[2]=c1*(h1r-data[2]);
			four1(data,n>>1,-1);
		}
}

void cosft1(float y[], int n)
{
		void realft(float data[], unsigned long n, int isign);
		int j,n2;
		float sum,y1,y2;
		double theta,wi=0.0,wpi,wpr,wr=1.0,wtemp;

		theta=PI/n;
		wtemp=sin(0.5*theta);
		wpr = -2.0*wtemp*wtemp;
		wpi=sin(theta);
		sum=0.5*(y[1]-y[n+1]);
		y[1]=0.5*(y[1]+y[n+1]);
		n2=n+2;
		for (j=2;j<=(n>>1);j++) {
			wr=(wtemp=wr)*wpr-wi*wpi+wr;
			wi=wi*wpr+wtemp*wpi+wi;
			y1=0.5*(y[j]+y[n2-j]);
			y2=(y[j]-y[n2-j]);
			y[j]=y1-wi*y2;
			y[n2-j]=y1+wi*y2;
			sum += wr*y2;
		}
		realft(y,n,1);
		y[n+1]=y[2];
		y[2]=sum;
		for (j=4;j<=n;j+=2) {
			sum += y[j];
			y[j]=sum;
		}
}
#undef PI
double eval(str, length, vect, genes)
char str[];      /* string representaticn                  */
int length;      /* length of bit string                   */
double vect[];          /* floating point representation         */
int genes;       /* number of elements in vect                   */
{
```

```
static int startup=1,mode,alpha,res;
static float toler=0.9;
FILE *tfile;
float *v;
double norm;
unsigned n,i,j;
int st;
double fitness=0.0,level,stfunc,dev,max,sum;
     if(startup)
     {
      startup=0;
      tfile=fopen("tol","rt");
      if(tfile!=NULL)
      {
         scanf(tfile,"%i%i%f%i",&mode,&alpha,&toler,&res);
      }
      fclose(tfile);
     }
     n=length;
     for(j=0;n>1;n>>=1,j++);
     n<<=j;
     if ((length-n>=8)&&(mode>2))
     {
      alpha=0;

      for(i=n-1;i<n+8;i++)
      {
         alpha<<=1;
         alpha+=(str[i]=='1');
      }
     }
     st=n/32+1;
     v=(float *) calloc(4,n+2);
     for(i=0; i<n; i++) {
           v[i+1] =2.0*(str[i]=='1')-1.0;
     }
     cosft1(v,n);  /* compute spectrum entire data set*/
     norm=0.0;
     max=0.0;
     for(i=1;i<=n;i++)
     {
      v[i]*=v[i];
      norm+=v[i];
     }
     if(norm>1.0e-10)
      {for(i=1;i<=n;v[i++]/=2*norm);}
```

```
 else
  {for(i=1;i<=n;v[i++]=0);}
 level=toler/(st+0.5);
 fitness=0.0;
 max=0.0;
 sum=0.0;
 for(i=1;i<st+1;i++)
 {
  if ((i<st)&&(i%2!=0))
  {
      stfunc=level;
      dev=(stfunc-v[i]);
      dev*=dev;
      if(dev>=max)
       {max=dev;};
      sum+=dev;
  }
  else
  {
      stfunc=0.0;
  }
 }
 if((mode%2)==0)
  {fitness=(alpha/256.0)*max+(1.0-alpha/256.0)*sum/st;}
 else
 {
  if(random(256)<alpha)
     {fitness=max;}
  else
     {fitness=sum/st;};
 };
 free(v);          /* free v[]          */
 return fitness;
}
```

```
/* This  PCFFT.C  program  by J.G.G. Dobbe -- Performs an FFT on two arrays
(Re, Im) of type float. */

/* --------------------- Include directive ------------------------ */

#include "pcfft.h"
#include <stdio.h>
#include <stdlib.h>

/* --------------------- Local variables -------------------------- */

static float CosArray[28] =

{ /* cos{-2pi/N} for N = 2, 4, 8, ... 16384 */

 -1.00000000000000,  0.00000000000000,  0.70710678118655,

  0.92387953251129,  0.98078528040323,  0.99518472667220,

  0.99879545620517,  0.99969881869620,  0.99992470183914,

  0.99998117528260,  0.99999529380958,  0.99999882345170,

  0.99999970586288,  0.99999992646572,

 /* cos{2pi/N} for N = 2, 4, 8, ... 16384 */

 -1.00000000000000,  0.00000000000000,  0.70710678118655,

  0.92387953251129,  0.98078528040323,  0.99518472667220,

  0.99879545620517,  0.99969881869620,  0.99992470183914,

  0.99998117528260,  0.99999529380958,  0.99999882345170,

  0.99999970586288,  0.99999992646572

};

static float SinArray[28] =

{ /* sin{-2pi/N} for N = 2, 4, 8, ... 16384 */

  0.00000000000000, -1.00000000000000, -0.70710678118655,
```

```
  -0.38268343236509, -0.19509032201613, -0.09801714032956,
  -0.04906767432742, -0.02454122852291, -0.01227153828572,
  -0.00613588464915, -0.00306795676297, -0.00153398018628,
  -0.00076699031874, -0.00038349518757,
  /* sin{2pi/N} for N = 2, 4, 8, ... 16384 */
   0.00000000000000,  1.00000000000000,  0.70710678118655,
   0.38268343236509,  0.19509032201613,  0.09801714032956,
   0.04906767432742,  0.02454122852291,  0.01227153828572,
   0.00613588464915,  0.00306795676297,  0.00153398018628,
   0.00076699031874,  0.00038349518757
};

/* --------------------- Function implementations ----------------- */
/* --------------------- ShuffleIndex ---------------------------- */
static unsigned int ShuffleIndex(unsigned int i, int WordLength)

/* Function     : Finds the shuffle index of array elements. The array length
                  must be a power of two; The power is stored in "WordLength".
   Return value : With "i" the source array index, "ShuffleIndex"
                  returns the destination index for shuffling.
   Comment      : -
*/
{
```

```
  unsigned int  NewIndex;
  unsigned char BitNr;
  NewIndex = 0;
  for (BitNr = 0; BitNr <= WordLength - 1; BitNr++)
  {
    NewIndex = NewIndex << 1;
    if ((i & 1) != 0) NewIndex = NewIndex + 1;
    i = i >> 1;
  }
  return NewIndex;
}
/* --------------------- Shuffle2Arr ------------------------------ */
static void Shuffle2Arr(float *a, float *b, int bitlength)
/* Function     : Shuffles both arrays "a" and "b". This function is called
                  before performing the actual FFT so the array elements
                  are in the right order after FFT.
  Return value : -
  Comment      : -
*/
{
  unsigned int  IndexOld, IndexNew;
  float         temp;
  unsigned int  N;
```

```
int bitlengthtemp;

bitlengthtemp = bitlength;          /* Save for later use */
N = 1;                              /* Find array-length */
do
{
 N = N * 2;
 bitlength = bitlength - 1;
} while (bitlength > 0) ;
                                    /* Shuffle all elements */
for (IndexOld = 0; IndexOld <= N - 1; IndexOld++)
{                                   /* Find index to exchange elements */
 IndexNew = ShuffleIndex(IndexOld, bitlengthtemp);
 if (IndexNew > IndexOld)
 {                                  /* Exchange elements: */
  temp = a[IndexOld];               /* Of array a */
  a[IndexOld] = a[IndexNew];
  a[IndexNew] = temp;
  temp = b[IndexOld];               /* Of array a */
  b[IndexOld] = b[IndexNew];
  b[IndexNew] = temp;
 }
}
```

```
}
/* --------------------- Fft ------------------------------------- */
void Fft(float *Re, float *Im, int Pwr, int Dir)

/* Function     : Actual FFT algorithm. "Re" and "Im" point to start of real
                  and imaginary arrays of numbers, "Pwr" holds the array sizes
                  as a power of 2 while "Dir" indicates whether an FFT (Dir>=1)
                  or an inverse FFT must be performed (Dir<=0).
  Return value : The transformed information is returned by "Re"
                  and "Im" (real and imaginary part respectively).
  Comment      : -
*/
{
  int pwrhelp;
  int N;
  int Section;
  int AngleCounter;
  int FlyDistance;
  int FlyCount;
  int index1;
  int index2;
  float tempr, tempi;
  float Re1, Re2, Im1, Im2;
```

```
float c, s;
float scale;
float sqrtn;
float temp;
float Qr, Qi;

Shuffle2Arr(Re, Im, Pwr);            /* Shuffle before (i)FFT */
pwrhelp = Pwr;                       /* Determine size of arrs */
N = 1;
do
{
  N = N * 2;
  pwrhelp--;
} while (pwrhelp > 0) ;

if (Dir >= 1) AngleCounter = 0;                 /* FFT */
else        AngleCounter = 14;              /* Inverse FFT */
Section = 1;
while (Section < N)
{
  FlyDistance = 2 * Section;
  c = CosArray[AngleCounter];
  s = SinArray[AngleCounter];
```

```
Qr = 1; Qi = 0;

for (FlyCount = 0; FlyCount <= Section - 1; FlyCount++)

{

 index1 = FlyCount;

 do

 {

  index2 = index1 + Section;

                         /* Perform 2-Point DFT */

  tempr = 1.0 * Qr * Re[index2] - 1.0 * Qi * Im[index2];

  tempi = 1.0 * Qr * Im[index2] + 1.0 * Qi * Re[index2];


  Re[index2] = Re[index1] - tempr;         /* For Re-part */

  Re[index1] = Re[index1] + tempr;

  Im[index2] = Im[index1] - tempi;         /* For Im-part */

  Im[index1] = Im[index1] + tempi;


  index1 = index1 + FlyDistance;

 } while (index1 <= (N - 1));


/*           k                       */

/* Calculate new Q = cos(ak) + j*sin(ak) = Qr + j*Qi */

/*       -2*pi                       */

/* with: a = -----                   */
```

```
    /*          N                          */
     temp = Qr;
     Qr  = Qr*c - Qi*s;
     Qi  = Qi*c + temp*s;
   }
   Section = Section * 2;
   AngleCounter = AngleCounter + 1;
  }
  if (Dir <= 0)                          /* Normalize for */
  {                                  /* inverse FFT only */
   scale = 1.0/N;
   for (index1 = 0; index1 <= N - 1; index1++)
   {
     Re[index1] = scale * Re[index1];
     Im[index1] = scale * Im[index1];
   }
  }
}
/* ----------------------------------------------------------------- */
```

(f_12.c new 4-27-94)

```
#include "extern.h"

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include "pcfft.h"

double f_12(str, length)

char str[]; /* string representation           */

int length; /* length of bit string            */

{

static int startup=1,mode=0,alpha=128,skip=4;

static float tol=0.7;

FILE *tfile;

float *v,*Re,*Im;

double norm;

unsigned n,i,j,maxzt,zt;

int st,tog=1;

double fitness=0.0,level,stfunc,dev,max,sum;

/*    if(startup)

    {

     startup=0;

     tfile=fopen("tol","rt");

     if(tfile!=NULL)

     {

      fscanf(tfile,"%i%i%f",&mode,&alpha,&tol);

     }

     fclose(tfile);
```

```
}
*/
n=length;
for(j=0;n>1;n>>=1,j++);
n<<=j;
if ((length-n>=8)&&(mode>=2))
{
 alpha=0;
 for(i=n-1;i<n+8;i++)
 {
alpha<<=1;
alpha+=(str[i] ? 1 : 0);
 }
}
st=n/8+1;
maxzt=2*st+2;
v=(float *) calloc(4,n+2);
Re=(float *) calloc(4,n+2);
Im=(float *) calloc(4,n+2);
zt=0;
tog=1;
for(i=0; i<n; i++)
{
 tog*=((str[i]) ? 1 : -1);
 v[i]=1.0*tog;
 if(i!=0)
 {
```

```
  if(v[i-1]!=v[i]) {zt++;}
 }
 Re[i]=v[i];
 Im[i]=0.0;
}
if(v[0]!=v[n-1]) {zt++;}
fft(Re,Im,j,1); /* compute spectrum entire data set*/
norm=0.0;
max=0.0;
for(i=0;i<n;i++)
{
 v[i]=Re[i]*Re[i]+Im[i]*Im[i];
 norm+=v[i];
}
if(norm>1.0e-10)
 {for(i=0;i<n;v[i++]/=norm);}
else
 {for(i=0;i<n;v[i++]=0.0);}
level=1.0*tol*skip/(2*st-1);
fitness=0.0;
max=0.0;
sum=0.0;
for(i=0;i<st;i++)
{
 if ((i<st)&&(i%skip))
 {
stfunc=level;
```

```
    dev=(stfunc-v[i]);
    dev*=dev;
    if(dev>=max)
     {max=dev;};
    sum+=dev;
      }
      else
      {
    stfunc=0.0;
      }
     }

     if((mode%2)==0)
      {fitness=(alpha/256.0)*max+(1.0-alpha/256.0)*sum/st;}
     else
     {
      if(random()*256.0<alpha*1.0)
    {fitness=max;}
      else
    {fitness=sum/st;};
     };
     free(Re);
     free(Im);
     free(v);        /*    free v[]      */
     if(zt>maxzt) {fitness+=fabs((zt-maxzt)*1.0e-8);}
     return fitness;
}
```

```
(powspec.c 4-27-95)
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "extern.h"
#include "pcfft.h"
double f_12x(str, length)
char str[]; /* string representation          */
int length; /* length of bit string           */
{
float *v,*w,*Re,*Im;
FILE *outfile;
double norm;
unsigned n,i,j;
int st,tog=1,skip=4;
double fitness=0.0,level,stfunc,dev,max,sum;
    n=length;
    for(j=0;n>1;n>>=1,j++);
    n<<=j;
    st=n/32+1;
    v=(float *) calloc(4,n+2);
    w=(float *) calloc(4,n+2);
    Re=(float *) calloc(4,n+2);
    Im=(float *) calloc(4,n+2);
    for(i=0; i<n; i++) {
    tog*=2*(str[i]=='1')-1;
    v[i]=1.0*tog;
```

```
Re[i]=v[i];
Im[i]=0.0;
}
fft(Re,Im,j,1); /* compute spectrum entire data set*/
norm=0.0;
for(i=0;i<n;i++)
{
w[i]=Re[i]*Re[i]+Im[i]*Im[i];
norm+=w[i];
}
level=1.0*skip/(2*st-1);

if(norm>1.0e-10)
{for(i=0;i<n;w[i++]/=1.0*norm);}
else
{for(i=0;i<n;w[i++]=0);}
outfile=fopen("powspec.dat","w+");
for(i=n/2;i<n;i++)
{
if (((i<st)| | (i>(n-st))&&(i%skip))
{
stfunc=level;
}
else
{
stfunc=0.0;
}
```

```
fprintf(outfile,"%4i %f %f %f \n",i-n,stfunc,w[i],v[i]);
}

for(i=0;i<n/2;i++)
{
if (((i<st)| | (i>(n-st))&&(i%skip))
{
stfunc=level;
}
else
{
stfunc=0.0;
}
fprintf(outfile,"%4i %f %f %f \n",i,stfunc,w[i],v[i]);
}
fclose(outfile);

free(Re);
free(Im);
free(w);
free(v);          /* free v[]        */
return 0.0;
}

main(int argc,char ** argv)
{
unsigned nv=128,j;
double fit=0.0;
char str[1024]=" ",tstr[10]="Length",Infile[10],Minfile[10],c;
```

```
FILE *stats,*data;
    if (argc < 2)
    {
     strcpy(Infile,"in");
     strcpy(Minfile,"min");
    }
    else
    {
    sprintf(Infile, "in.%s", argv[1]);
    sprintf(Minfile, "min");
    }
    stats=fopen(Infile,"rt");
    data=fopen(Minfile,"rt");
    if(stats!=NULL)
    {
     while(0!=strcmp(tstr,str))
  {fscanf(stats,"%s",str);}
     fscanf(stats,"%s",str);
     fscanf(stats,"%s",str);
     fscanf(stats,"%s",str);
     fscanf(stats,"%i",&nv);
     fclose(stats);
     j=0;
     while((fscanf(data,"%c",&c))&&(j<nv))
     {
  if((c=='1')||(c=='0'))
```

```
    {str[j++]=c;}
     }
     str[j]='\0';
     fclose(data);
     f_12x(str,nv);
    }
    else
    {
     printf("invalid extension\n");
    }
    return 0;
}
```

```
/* This program prints out a single cell of a one or two dimensional Dammann
Gratings *\

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define PI 3.141592653589793
#define SWAP(a,b) tempr=(a);(a)=(b);(b)=tempr

void four1(float data[], unsigned long nn, int isign)
{
	unsigned long n,mmax,m,j,istep,i;
	double wtemp,wr,wpr,wpi,wi,theta;
	float tempr,tempi;

	n=nn << 1;
	j=1;
	for (i=1;i<n;i+=2) {
		if (j > i) {
			SWAP(data[j],data[i]);
			SWAP(data[j+1],data[i+1]);
		}
		m=n >> 1;
		while (m >= 2 && j > m) {
			j -= m;
			m >>= 1;
		}
		j += m;
	}
	mmax=2;
	while (n > mmax) {
		istep=mmax << 1;
		theta=isign*(6.28318530717959/mmax);
		wtemp=sin(0.5*theta);
		wpr = -2.0*wtemp*wtemp;
		wpi=sin(theta);
		wr=1.0;
		wi=0.0;
		for (m=1;m<mmax;m+=2) {
			for (i=m;i<=n;i+=istep) {
				j=i+mmax;
				tempr=wr*data[j]-wi*data[j+1];
				tempi=wr*data[j+1]+wi*data[j];
				data[j]=data[i]-tempr;
				data[j+1]=data[i+1]-tempi;
				data[i] += tempr;
```

```
                    data[i+1] += tempi;
                }
                wr=(wtemp=wr)*wpr-wi*wpi+wr;
                wi=wi*wpr+wtemp*wpi+wi;
            }
            mmax=istep;
        }
}
#undef SWAP

void realft(float data[], unsigned long n, int isign)
{
        void four1(float data[], unsigned long nn, int isign);
        unsigned long i,i1,i2,i3,i4,np3;
        float c1=0.5,c2,h1r,h1i,h2r,h2i;
        double wr,wi,wpr,wpi,wtemp,theta;

        theta=3.141592653589793/(double) (n>>1);
        if (isign == 1) {
                c2 = -0.5;
                four1(data,n>>1,1);
        } else {
                c2=0.5;
                theta = -theta;
        }
        wtemp=sin(0.5*theta);
        wpr = -2.0*wtemp*wtemp;
        wpi=sin(theta);
        wr=1.0+wpr;
        wi=wpi;
        np3=n+3;
        for (i=2;i<=(n>>2);i++) {
                i4=1+(i3=np3-(i2=1+(i1=i+i-1)));
                h1r=c1*(data[i1]+data[i3]);
                h1i=c1*(data[i2]-data[i4]);
                h2r = -c2*(data[i2]+data[i4]);
                h2i=c2*(data[i1]-data[i3]);
                data[i1]=h1r+wr*h2r-wi*h2i;
                data[i2]=h1i+wr*h2i+wi*h2r;
                data[i3]=h1r-wr*h2r+wi*h2i;
                data[i4] = -h1i+wr*h2i+wi*h2r;
                wr=(wtemp=wr)*wpr-wi*wpi+wr;
                wi=wi*wpr+wtemp*wpi+wi;
        }
        if (isign == 1) {
                data[1] = (h1r=data[1])+data[2];
```

```
			data[2] = h1r-data[2];
		} else {
			data[1]=c1*((h1r=data[1])+data[2]);
			data[2]=c1*(h1r-data[2]);
			four1(data,n>>1,-1);
		}
}

void cosft1(float y[], int n)
{
	void realft(float data[], unsigned long n, int isign);
	int j,n2;
	float sum,y1,y2;
	double theta,wi=0.0,wpi,wpr,wr=1.0,wtemp;

	theta=PI/n;
	wtemp=sin(0.5*theta);
	wpr = -2.0*wtemp*wtemp;
	wpi=sin(theta);
	sum=0.5*(y[1]-y[n+1]);
	y[1]=0.5*(y[1]+y[n+1]);
	n2=n+2;
	for (j=2;j<=(n>>1);j++) {
		wr=(wtemp=wr)*wpr-wi*wpi+wr;
		wi=wi*wpr+wtemp*wpi+wi;
		y1=0.5*(y[j]+y[n2-j]);
		y2=(y[j]-y[n2-j]);
		y[j]=y1-wi*y2;
		y[n2-j]=y1+wi*y2;
		sum += wr*y2;
	}
	realft(y,n,1);
	y[n+1]=y[2];
	y[2]=sum;
	for (j=4;j<=n;j+=2) {
		sum += y[j];
		y[j]=sum;
	}
}
#undef PI

double eval(str, length)
char str[];	/* string representation				*/
int length;	/* length of bit string				*/
{
float *v,*g;
```

```
unsigned n,i,j,ia,ja;
double norm;
int byte;
FILE *imagef;

    n=length;
    for(j=0;n>2;n>>=1,j++);
    n<<=j;
    v=(float *) calloc(4,n+2);
    g=(float *) calloc(4,n+2);
    for(i=0; i<n; i++) {
        v[i+1] = 2.0*(str[i]=='1')-1.0;
        g[i+1] = v[i+1];

    }
    cosft1(v,n);  /* compute spectrum entire data set*/
    norm=0;
    for(i=1;i<=n;i++)
    {
     v[i]*=v[i];
     if(norm<v[i])
        norm=v[i];
    }
    for(i=1;i<=n;v[i++]/=norm);
    imagef=fopen("dimage.ps","wt");
    fputs("%!PS-Adobe-2.0\n",imagef);
    fputs("gsave\n",imagef);
    fputs("initgraphics\n",imagef);
    fputs("0 0 translate\n",imagef);
    fputs("0.24 0.24 scale\n",imagef);
    fprintf(imagef,"/imline %d string def\n",n*2);
    fputs("/drawimage {\n",imagef);
    fprintf(imagef,"%d %d 8 \n",n,n);
    fprintf(imagef,"[%d 0 0 %d 0 %d]\n",n,-1*n,n);
    fputs("{currentfile imline readhexstring pop} image\n",imagef);
    fputs("} def\n",imagef);
    fputs("2550 2550 scale\n",imagef);
    fputs("drawimage\n",imagef);
    for(i=0;i<n;i++)
    {
     for(j=0;j<n;j++)
     {
        ia=i+1;
        ja=i+1;
        byte=(short) (255*(1-v[ia]*v[ja]));
        fprintf(imagef,"%02X",byte);
```

```
      }
      fputs("\n",imagef);
     }
     fputs("showpage\ngrestore\n",imagef);
     fclose(imagef);
     free(v);          /* free v[]        */
     free(g);
     return 0.0;
}

grate(char str[],int len)
{
 int i,j,nb,byte,nbyte;
 char line1[1024]="",line0[1024]="",bite[4]="";
 FILE *imagef;

 nb=len/8;
 for(i=0;i<nb;i++)
 {
  byte=0;
  nbyte=0;
  for(j=0;j<8;j++)
  {
   byte|=(str[i*8+j]=='1')<<(7-j);
   nbyte|=(str[i*8+j]=='0')<<(7-j);
  }
  sprintf(bite,"%02x",byte);
  strcat(line0,bite);
  sprintf(bite,"%02x",nbyte);
  strcat(line1,bite);
 }
 strcat(line0,"\n");
 strcat(line1,"\n");
 imagef=fopen("image.ps","wt");
 fputs("%!PS-Adobe-2.0\n",imagef);
 fputs("/pixbuf 2 string def \n",imagef);
 fputs("gsave\n",imagef);
 fputs("/Helvetica findfont 24 scalefont setfont\n",imagef);
 fputs("172 236 moveto\n",imagef);
 fputs("(Dammann Grating) show\n",imagef);
 fputs("172 272 translate\n",imagef);
 fputs("256 256 scale\n",imagef);
 fprintf(imagef,"%i %i 1 [%i 0 0 %i 0 0] \n",len,len,len,len);
 fputs("{currentfile pixbuf readhexstring pop} image\n",imagef);
 for(i=0;i<len;i++)
  if(str[i]=='1')
```

```
  {fputs(line1,imagef);}
 else
  {fputs(line0,imagef);}
 fputs("showpage\ngrestore\n",imagef);
 fclose(imagef);
}

grater(char str[],int len)
{
 int i,j,toggle,trans,x1,y1,x2,y2;
 char tog;
 int line[1024];
 FILE *imagef;
 trans=1;
 tog=str[0];
 line[0]=0;
 for(i=0;i<len;i++)
 {
  if(tog!=str[i])
  {
   tog=str[i];
   line[trans++]=i;
  }
 }
 line[trans]=len;
 imagef=fopen("rimage.ps","wt");
 fputs("%!PS-Adobe-2.0\n",imagef);
 fputs("/Helvetica findfont 24 scalefont setfont\n",imagef);
 fputs("172 236 moveto\n",imagef);
 fputs("(Dammann Grating) show\n",imagef);
 fputs("172 272 translate\n",imagef);
 fputs(".5 .5 scale\n",imagef);
 fputs("0 setgray \n",imagef);
 fputs("/r { /y1 exch def /x1 exch def /y2 exch def /x2 exch def \n",imagef);
 fputs("x1 y1 moveto\n",imagef);
 fputs("x1 y2 lineto\n",imagef);
 fputs("x2 y2 lineto\n",imagef);
 fputs("x2 y1 lineto\n",imagef);
 fputs("closepath fill } def\n",imagef);
 fputs("/tmask { /tar exch def\n",imagef);
 fputs("0 1 tar length 2 sub { /i exch def 0 1 tar length 2 sub {\n",imagef);
 fputs("/j exch def i j add 2 mod 1 eq { tar i get tar j get \n",imagef);
 fputs("tar i 1 add get tar j 1 add get r\n",imagef);
 fputs("} if } for } for } def\n",imagef);
 fputs("\n [ ",imagef);
 for(i=0;i<trans;i++)
```

```
 {
  x1=line[i];
  fprintf(imagef,"%i ",x1);
 }
 fprintf(imagef,"%i ] tmask \n showpage\n",len);
 fclose(imagef);
}


main(int argc,char ** argv)
{
unsigned nv=128,j;
double fit=0.0;
char str[1024]=" ",tstr[10]="Length",Infile[10],Minfile[10],imfile[10],c;
FILE *stats,*data,*imageg;
    if (argc < 2)
    {
     strcpy(Infile,"in");
     strcpy(Minfile,"min");
    }
    else
    {
    sprintf(Infile, "in.%s", argv[1]);
    sprintf(Minfile, "min.%s", argv[1]);
    }
    stats=fopen(Infile,"rt");
    data=fopen(Minfile,"rt");
    if(stats!=NULL)
    {
     while(0!=strcmp(tstr,str))
        {fscanf(stats,"%s",str);}
     fscanf(stats,"%s",str);
     fscanf(stats,"%i",&nv);
     for(j=0;nv>1;nv>>=1,j++);
     nv<<=j;
     fclose(stats);
     j=0;
     while((fscanf(data,"%c",&c))&&(j<nv))
     {
        if((c=='1')||(c=='0'))
         {
          str[j+nv]=c;
          str[nv-j-1]=c;
          j++;
         }
     }
```

```
     str[j+nv]='\0';
     fclose(data);
     grate(str,nv*2);
     grater(str,nv*2);
     eval(str,nv);
    }
    else
    {
     printf("invalid extension\n");
    }
    return 0;
}
```

LIST OF REFERENCES

[1] S.A Collins, Jr. and H.J. Caulfield. "Optical holographic interconnects: categorization and potential efficient passive resonated holograms." J.Opt.Soc. Am A/Vol.6,No.10 1989.

[2] H.M. Ozaktas and J.W. Goodman. "Implications of interconn- ection theory for optical digital computing." Applied Optics, Vol.31, No.26 p.5559 1992.

[3] C. Tocci and H. J. Caulfield. "Optical Interconnection Foundations and Applications". Artech House, Inc., 1994

[4] H. H. Arsenault, and Y. Sheng. "An Introduction to Optics in Computers". The International Society of Optical Engineering, p.54, 1992.

[5] J. Jahns, M.M. Downs, M.E. Prise, N.Steibl,and S.J. Walker. "Dammann Gratings for Laser beam shaping." Optical Engineering, Vol.28,No.12, p.1267, 1989.

[6] H. Dammann and K. Gortler. "High-Efficiency In-Line Multiple Imaging by means of Multiple Phase Holograms." Optics Comm. Vol.3 No.5 p.312 1971.

[7] J.N. Mait. "Design of binary-phase and multiphase Fourier gratings for array generation." Opt. Soc. Am. A/Vol.7, No.8 p.1514 1990.

[8] U. Krackhardt and N. Streibl. "Design of Dammann Gratings for array generation." Optics Comm., Vol.74 No.1,2 p.31 1989.

[9] D.G. Feitelson. "Optical Computing"

[10] S.J. Walker and J. Jahns. "Optical clock distribution using integrated free- space optics." Optics Comm., Vol.90 p.359 1992.

[11] D.E. Goldberg. "Genetic Algorithms in Search, Optimization, and Machine Learning." Addison-Wesley, Reading Mass., 1987.

[12] F. Hoffmeister and T. Back. "Genetic Self-Learning." University of Dortmund, Dept. of Computer Science.

[13] J. H. Holland. "Adaptation in Natural Artificial Systems." The University of Michigan Press, Ann Arbor, 1975.

[14] J.Turunen, A.Vasara, J.Westerholm, G. Jin and A.Salin. "Optimisation and fabrication of grating beamsplitters." J.Physics,D21, S102,1988.

[15] F.B. McCormick. "Generation of large spot arrays from a single laser beam by multiple imaging with binary phase gratings." Optical Engineering, Vol.28 No.4 p.299, 1989.

[16] U.Efron, A.D. Fisher, and C. Warde. "Spatial Light Modulator for Optical Information Processing." Applied Optics, Vol.28, p.4739 1989.

[17] A.D. Fisher and J.N. Lee. "The current status of two dimensional Spatial Light Modulator technology." SPIE Proc. Vol.634, p.352, 1986.

[18] D.Armitage, J.I. Thackara, and W.D.Eades. "Photoaddressed Liquid Crystal Spatial Light Modulators." Applied Optics Vol.28,No.22, 1989.

[19] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vettering. "Numerical Recipes in C". The Art of Scientific Computing. Cambridge University Press Cambridge. P.398, 1988, First Edition.

[20] H. Lupken, T. Pauka, R.Brauer, F. Wyrowski and O. Bryngdahl. "On the design of Dammann Gratings." Optics Communication No. 100 1993 pages 415-420.

[21] C.H. Kwak, S.Y. Park, H.M. Kim, El-Hang Lee, and C.M. Kim. "Dammann gratings for Multispot array generation by using photoinduced anisotropic materials." Optics Communications No. 88 1992 pages 249-257.

[22] M. R. Taghizadeh and J. Turunen. "Synthetic diffractive elements for optical interconnection." Optical Computing and Processing Vol.2 No. 4 1992, pages 221-242.

[23] U. Mahlab, J. Shamir, and H.J. Caulfield. "Genetic algorithm for optical pattern recognition." Optics Letters, vol. 16, No.9 1991 pages 648- 650.

[24] J. Jahns, M. M. Downs, M. E. Prise, N. Streibl, and S.J. Walker. "Dammann gratings for laser beam shaping." Optical Engineering, Vol.28, No.12 1989, pages 1267-1275.

[25] M. R. Taghizadeh, J. I. B. Wilson, J. Turunen, A. Vasara, and J. Westerholm "Optimization and Fabrication of grating beamsplitters in Silicon Nitride". Applied Physics Letters, Vol. 54, No.16, 1989, pages 1492 -1494.

[26] L. Davis "Genetic Algorithms and Simulated Annealing". Kaufmann Publishers, 1987.

[27 J. D. Gaskill, "Linear Systems, Fourier Transforms, and Optics". New York, Wiley, 1978.

☆U.S. GOVERNMENT PRINTING OFFICE: 1997-509-127-47171

MISSION
OF
ROME LABORATORY

Mission. The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

a. Conducts vigorous research, development and test programs in all applicable technologies;

b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;

c. Provides a full range of technical support to Air Force Material Command product centers and other Air Force organizations;

d. Promotes transfer of technology to the private sector;

e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.