tional to Q_{l0}:

$$M_{l-1,0} = -\frac{\omega}{c}\frac{l-1}{[(2l+1)(2l-1)]^{1/2}}Q_{l0}. \quad (13)$$

[1]J. D. Jackson, *Classical Electrodynamics* (Wiley, New York, 1975), 2nd ed., Chap. 16.

[2]S. Datta, "Multipole expansion of the interaction hamiltonian between a charged particle and a non-uniform magnetic field," European J. Phys. **5**, 243–250 (1984).

[3]J. Bronzon, "Magnetic scalar potential," Am. J. Phys. **39**, 1357–1359 (1971).

[4]C. G. Gray, "Multipole expansions of electromagnetic fields using Debye potentials," Am. J. Phys. **46**, 169–179 (1978).

Optical transformations in three-space: Simulations with a PC

Raymond G. Wilson and Sean M. McCreary
Department of Physics, Illinois Wesleyan University, Bloomington, Illinois 61702

Jeffrey L. Thompson
IBM Corporation, Bloomington, Illinois 61701

(Received 4 September 1990; accepted 27 May 1991)

Using Fourier transform and convolution techniques, it is illustrated how all aspects of Fraunhofer diffraction of virtually any aperture real or complex can be calculated and displayed in two- and three-space using an IBM-compatible PC. Details of program commands using software MATLAB™ are illustrated as well as results of plotting software, SURFER™. Fresnel diffraction is illustrated using Fourier transformation.

I. INTRODUCTION

It is now possible to explore the Fourier transform relationships that result from the scalar theory of diffraction, in three-space, by personal computer. All aspects of Fraunhofer diffraction of essentially any two-dimensional aperture can be obtained rather simply: amplitude, irradiance (point spread function), modulation and phase transfer functions; real or complex. Additionally, some aspects of Fresnel diffraction may also be portrayed.

The above statement is intended to show contrast with past articles of this journal[1] in which, theoretically or experimentally, diffraction is often treated as one dimensional; or if two dimensional, we are presented with only irradiance pictures. We can now have it all. Optics journals have recently used some three-space computer graphics, but how they did it is only briefly described, if at all. We hope to remedy that here. Admittedly, one may need ingenuity creating the aperture matrix.

We have been using 286 and 386 computers and two IBM-compatible software packages, $\text{SURFER}_{\text{TM}}$[2] and $\text{MATLAB}_{\text{TM}}$.[3] If one can solve by hand for the Fraunhofer diffraction amplitude, e.g., the $z(x,y) = \text{Cab}\,\text{sinc}(\pi ax) \times \text{sinc}(\pi by)$ for the $a \times b$ rectangular aperture, then SURFER can plot this $z(x,y)$ very easily at any orientation and with great resolution of detail. All of our plots were drawn by a Hewlett–Packard 9872B (11″ × 17″ paper size) pen plotter. SURFER, in its Grid program, allows one to square the amplitude function and plot the irradiance or point spread function. If the Fraunhofer amplitude is mathematically complex, e.g., phase shifted, one can also have SURFER plot phase in the amplitude pattern, $\Theta(x,y) = \arctan(\text{Im}\,z/\,\text{Re}\,z)$. SURFER also allows one to do element-by-element operations between two grids of $z(x,y)$ data and hence create a third output grid of $z(x,y)$ data for plotting or for further manipulation and simulation.

But SURFER cannot do all of the mathematical operations required in the Fourier approach to diffraction. The convolution theorem as mapped out in Fig. 1 is a very convenient way to remember the Fourier approach. Examination of it will show it to be a complete description of Fraunhofer diffraction, and more. It is the MATLAB software, and other such utilities, which allow one to do: two-dimensional fast Fourier transformations,[4] inverse transformations, convolutions, correlations, etc., on data in matrix form. Many modern optics textbooks introduce Fourier techniques and it is possible that such an approach might be better in first year calculus-based General Physics rather than the limited, perhaps obscure, phasor treatment of diffraction.

II. THE SQUARE APERTURE

As an example of the ease of use we will first demonstrate diffraction by a single square aperture. PC-MATLAB allows

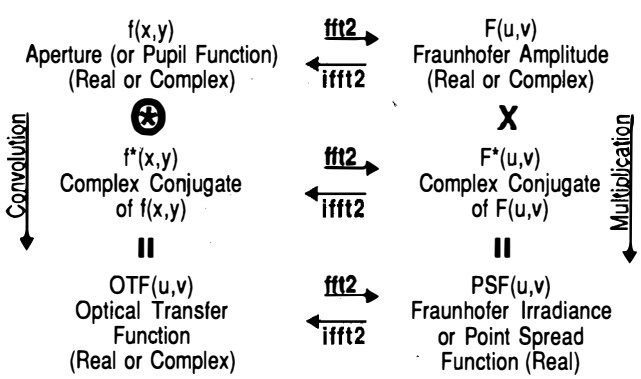

Fig. 1. A Fourier description of Fraunhofer diffraction. The OTF requires normalization for expression in (u,v) coordinates.

© 1991 American Association of Physics Teachers

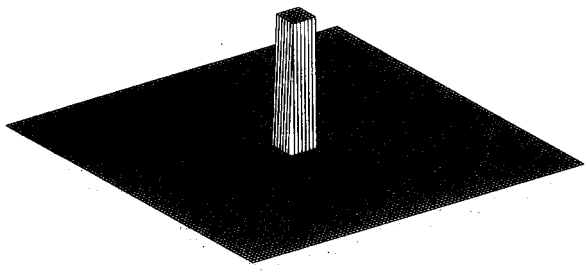

Fig. 2. The square aperture.

a 64×64 matrix working space; AT-MATLAB, which we have used here, allows a 90×90 matrix. Apparently, 386-MATLAB has no limit, but all have specific hardware requirements. Because of the reciprocal relationship between aperture size and spread of the Fourier transform, we find that a square of size 7×7 within the 90×90 matrix is a nice size for diffraction portrayal. We shall describe here each step for this square aperture but relegate subsequent details to an Appendix:

sq = zeros(90);	creates a 90×90 array of zeros. We chose the name sq.
sq(43:49,43:49) = ones(7);	creates the centered 7×7 square aperture; i.e., sq(*x*start *x*end,*y*start:*y*end).
mesh(sq)	yields Fig. 2
sqft = fft2(sq);	creates the fast Fourier transform.
mesh(sqft)	will display it.

It is sometimes referred to as the Amplitude Point Spread Function (ASF).[5] It will have an appearance somewhat like Fig. 3, "like" because they may also be a spiky nature from MATLAB's calculations. The **fft2** algorithm places the zero order at one corner of the matrix. Figure 3 is actually the real part of the transform, and each quadrant requires a 180-deg rotation.

To eliminate the spiky nature it seems best to first shift the square to the corners of the matrix, then do **fft2**. The command **fftshift**(sq); does that, Fig. 4. Figure 3 was mesh(fft2(fftshift(sq))). Much of this can be combined

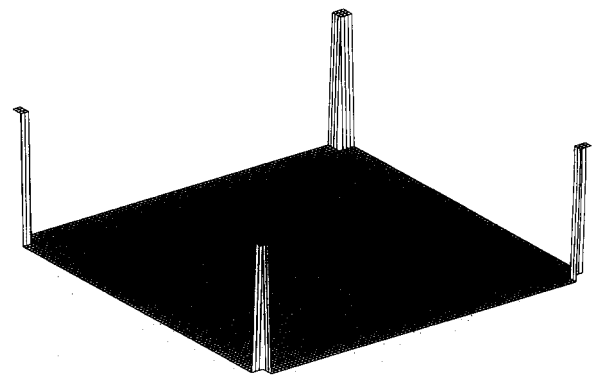

Fig. 4. The FFT shifted square.

into a single line, such as,

sqreft = real(fftshift(fft2(fftshift(sq))));

mesh(sqreft) yields Fig. 5,

the real part of the Fourier transform of the square aperture. Because some transformations are complex one might also wish to create the modulus

sqmdft = abs(ffshift(fft2(fftshift(sq))));

mesh(sqmdft) yields the modulus diagram.

Often one runs out of useable computer memory. With MATLAB the command, **whos**, will show what occupies memory. The command, **pack**, will rearrange it. Some few 100K bytes are used in each of the calculations just performed. The command, **clear** *filename*, followed by **pack** and **whos** will erase the named file, create more useable memory and show memory available. The command, **save** *filename*, will save the data to hard disk with the extension .mat, for reloading when needed again.

There was a print error in early versions of MATLAB preventing printout of 90×90 graphics. It has been corrected in later versions.

Our focus in this paper is on the optical functions, not on axis scaling or labeling. This can be done, as one chooses, in MATLAB and SURFER.

The Fraunhofer irradiance, also called the point spread function (PSF), can be displayed,

sqft = fftshift(fft2(fftshift(sq)));

creating the amplitude function. It may be complex, and will be, if the aperture, sq, is not centered in the matrix.

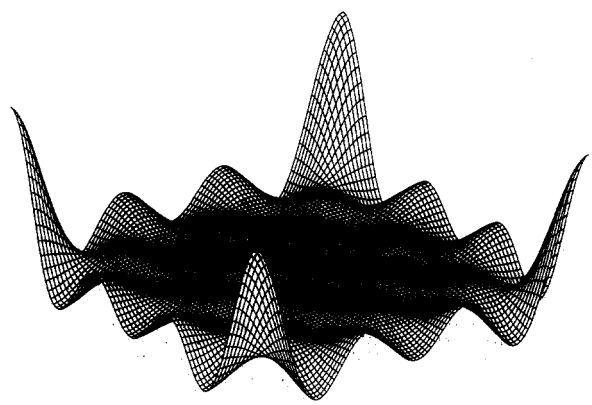

Fig. 3. The square aperture Fourier transform (modulus), zero order in the matrix corner.

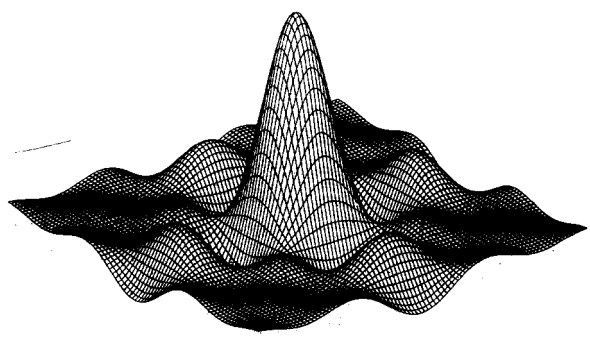

Fig. 5. The square aperture Fourier transform (real part), zero-order centered.

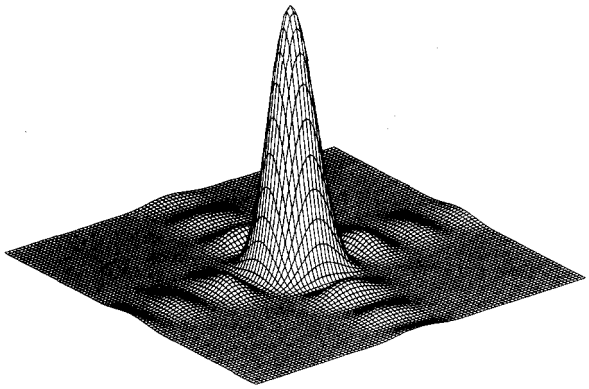

Fig. 6. The square aperture point spread function.

Recall that a Fraunhofer diffraction pattern is phase shifted if the symmetric aperture is not centered on the optical axis. The command **mesh**(sqft) will display only the real part. The irradiance then,

```
psf = sqft.*conj(sqft);    .*is element-by-element
                           multiplication.
mesh(psf)
```

will calculate and display the point spread function, Fig. 6.

We know that the adjacent regions of light along the axes in the diffraction pattern of a square are phase shifted by pi radians. Kaiser and Russel[6] suggest a nice way to demonstrate this. But diagonally in the pattern there are no phase shifts whatsoever.

```
sqftph = angle(sqft);
```

creates the phase matrix of the aperture Fourier transform, i.e., arctan(Imaginary part of ft/Real part of ft). The algorithm, **angle** usually does not do a splendid job of keeping track of 2π phase changes. The command **unwrap** attempts to smooth out such phase plots by removing branch cuts in the arctan function. Several unwraps may bring continued improvement. We created a MATLAB function called **unwrap2** which executes the command **unwrap**(**unwrap**(**angle**(*filename*))')' to speed up the operation, apostrophes yielding transposed matrices. Figure 7 shows phase in the Fourier transform unwrapped a few times. The central maximum has zero phase, all steps are π, and note the 2π or absence of phase change diagonally.

Completion of the operations in Fig. 1, bringing this application of the convolution theorem full circle, requires the calculation of the optical transfer function (OTF). The OTF is very useful when the aperture function is used for imaging with incoherent light. The OTF is made up of a modulation transfer function (MTF) and a phase transfer function (PTF); $\mathrm{OTF} = (\mathrm{MTF})e^{i(\mathrm{PTF})}$, where $\mathrm{MTF} = |\mathrm{OTF}|$ or modulus of the OTF. The MTF describes the contrast allowed each spatial frequency in an image plane, while the PTF describes phase shifts for each spatial frequency in an image. But a description of computer simulation of imaging in three-space using Fourier methods would be the subject of another paper for this journal.

The optical transfer function (OTF) can be calculated by two different paths according to Fig. 1: (1) By inverse Fourier transformation of PSF, (2) By discrete convolution of the aperture function, *slower* but more accurate. If we do this with the PSF from the 7×7 aperture we will get a rather small OTF surrounded by a mostly flat surface of zeros. A 20×20 square can be created for Method 1:

```
sq = zeros(90);
sq(37:56,37:56) = ones(20);
sqft = fftshift(fft2(fftshift(sq)));
psf = sqft.*conj(sqft);
otf = ifft2(psf);
mesh(otf)              and we see it in the corners,
                       quite spiky.
otf = fftshift(otf);
mesh(otf)              and small values of imaginary
                       parts may have created phase
                       noise. We know this OTF is
                       real.
otf = abs(otf);
mesh(otf)              would be a good portrayal,
                       though still surrounded by a
                       flat surface of zeros.
```

Having created the centered PSF as above, the OTF calculation can be condensed to a single line:

```
otf = fftshift(ifft2(fftshift(psf)));
```

creates OTF without phase noise. Method 1 is limited to 45×45 squares in the 90×90 array of AT-MATLAB.

Method 2:

```
otf = conv2(ones(20),ones(20));
mesh(otf)          will portray a 39×39 gridded OTF.
```

The maximum size:

```
otf = conv2(ones(45),ones(45));
mesh(otf)          yields an 89×89 gridded OTF, Fig.
                   8. It takes minutes.
```

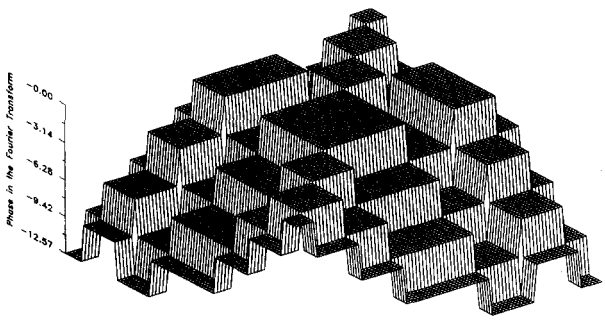

Fig. 7. The square aperture, phase in the Fourier transform.

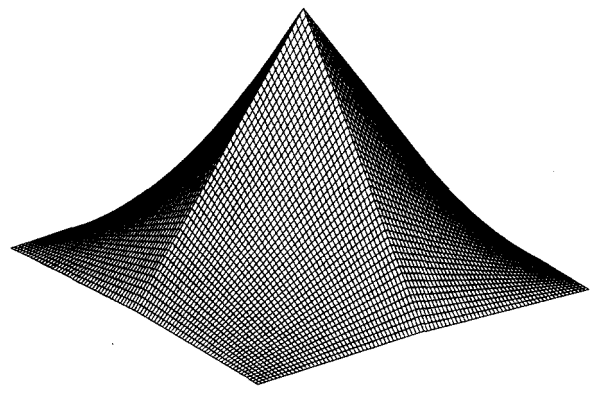

Fig. 8. A portrait of the square's OTF.

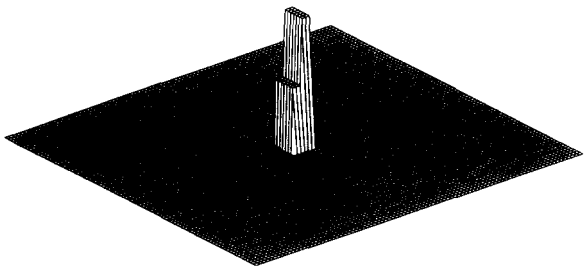

Fig. 9. Square aperture, one half with only one half amplitude transmittance.

MATLAB can output to a printer and figure tilt and rotation can be changed within MATLAB. We translated all MATLAB data files to SURFER binary Grid files and pen-plotted out of SURFER. SURFER as well as MATLAB can do topographic plots and SURFER can also stack a contour plot over the three-space surface plot. Topographic plots have not been included here for they will not nicely withstand size reduction for this Journal. They are available from author RGW.

III. OTHER REAL APERTURES

We are struck by the simplicity now available for three-space calculation and display to simulate Fourier diffraction techniques. We will now demonstrate a few more.

A. Half square, half transmitting

An interesting variation on the square aperture is the square aperture with half of it only partially transmitting. Using the same 7×7 square and reducing amplitude transmission by 1/2 in one-half the aperture we get Fig. 9, which is a good size for Fourier transforming. The modulus of the Fourier transform, Fig. 10, looks distinctly different. Phase changes are shown in Fig. 11 and topography of the modulus can also be shown. Comparison with the fully transmitting square is worthwhile, and further transmission reduction of half of the aperture, say to 1/4, leads one to what would be expected, thinking of reduction all the way to zero. In this case, zero, the aperture would no longer be symmetric and a linear phase shift will appear in one direction. The formulas can be calculated by hand, and SURFER used to get higher resolution than these 90×90 plots; 126×126 gives nice detail.

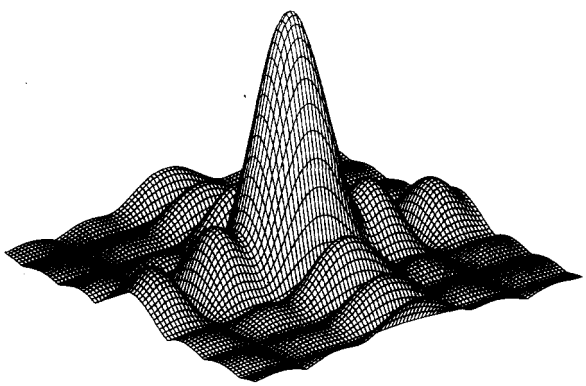

Fig. 10. Modulus of the Fourier transform of Fig. 9.

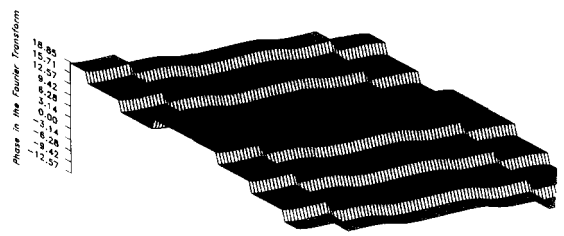

Fig. 11. Phase in the Fourier transform of Fig. 9.

B. Circular

Diffraction by a circular aperture can be demonstrated by the same procedures. It is impossible to get a perfectly round circle in a 90×90 square matrix of MATLAB, especially of small enough size with which to do good Fourier transforms; the ragged edge affects the transforms noticeably. We created a MATLAB function called **circ** (*radius*) to produce square-symmetric circular apertures. But SURFER understands Bessel functions of the first and second kind, of any order and will yield high resolution graphic simulations.

C. Square and circle

We created in MATLAB a double aperture, square and circle each of the same width, 24, spaced similarly. Figure 12 is the discrete convolution; which represents the optical transfer function for this double aperture. See the Appendix for MATLAB commands. It is clear from the OTF, achieved by convolution, that a folding operation occurred prior to the shifting process. Without folding, an autocorrelation would result, something quite different, Fig. 13. Anticipating optical correlations, MATLAB did this one simply. See the Appendix for commands.

D. A composite aperture

Diffraction by composite apertures can easily be simulated because of the linearity of Fourier transforms. The diffraction amplitude of an annular ring can be determined by subtracting the transform of the interior circle from that

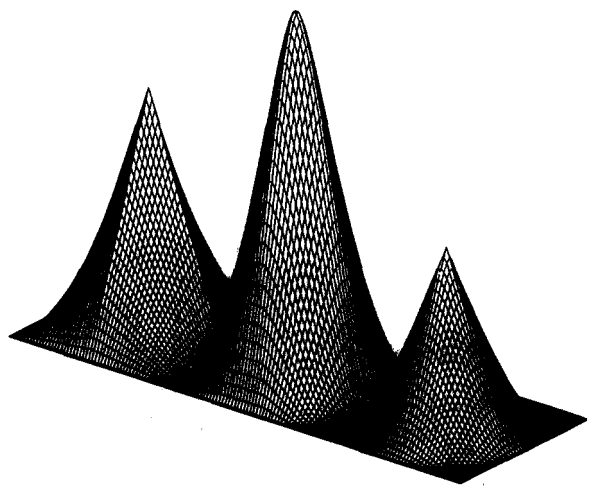

Fig. 12. The self-convolution of the double aperture.

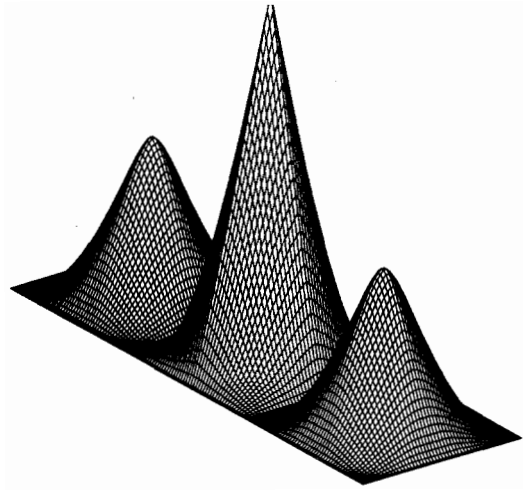

Fig. 13. The autocorrelation of the double aperture.

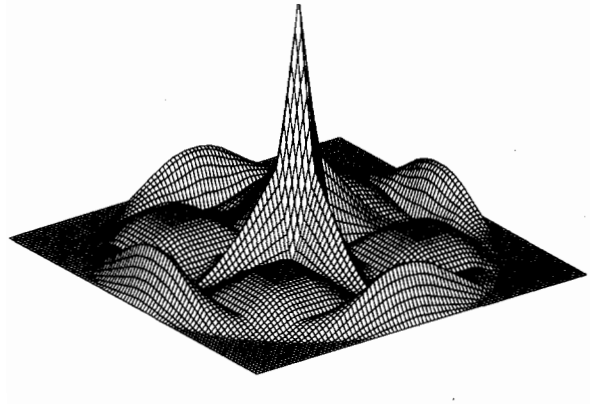

Fig. 15. The OTF of the aperture of Fig. 14.

of the exterior circle; diffraction by an annular square follows similarly. Since these calculations can be done by hand, SURFER can yield high resolution graphics from them. But using the 90×90 array of MATLAB Fig. 14 shows a circular aperture blocked by an opaque square of diagonal the same as the circle's diameter. Figure 15 shows the real optical transfer function obtained by discrete convolution of the aperture, a calculation one would not like to do by hand.

IV. INVERSE PROBLEMS

We can turn the process around, work it backwards. One can propose a real function to represent the Fourier transform of a hypothetical aperture. With this function entered into MATLAB, the following commands should yield the hypothetical aperture: hyp-ap = fftshift(ifft2(fftshift(*real function*))); mesh(hyp-ap).

The command for inverse Fourier transform is **ifft2**. This inverse process is easily demonstrated when you have the transform (real) for the square aperture sitting in the computer memory. The commands nicely return the square aperture.

If there is phase variation in the function or the aperture then one has a problem considerably more difficult, i.e., complex, to analyze. Recall that the phase information contained in a complex function is essentially lost when that function is squared. But even so, when dealing with complex functions, MATLAB can be asked to present the real part, the imaginary part, the modulus, or the phase contained therein. An example is available from author RGW.

When one looks at the x-ray diffraction pattern of, say, a biologically important molecule, the question is inverse also, "What structure caused that?" When the radar signal returns, bounced off the aircraft, the question is, "What aircraft is that?" Using lasers in diffraction and interference experiments of physics courses everyone knows 632.8 nm is the wavelength. Ask the students to fully determine the structure causing the diffraction pattern.

V. EDGE DETECTION BY SPATIAL FILTERING

An example of spatial filtering: Fig. 16 represents a square aperture blocked by an opaque triangle. We will attempt to do edge detection. The information about sharp edges is in the high spatial frequencies of the Fourier transforms. We create the transform matrix then go into it and block out (set to zero) a square array of all the low spatial frequency terms. Figure 17 shows the modulus of the modified transform. Now Fourier transform this modified transform matrix, which is equivalent to reimaging the aperture with another lens. What one would see or detect is the complex square of this transform, Fig. 18, wherein we would see brightness mainly along the edges of the square aperture and its blocking triangle. We think this is rather neat! Have you ever tried demonstrating spatial filtering to a large class? These images can be used as slides or overhead transparencies.

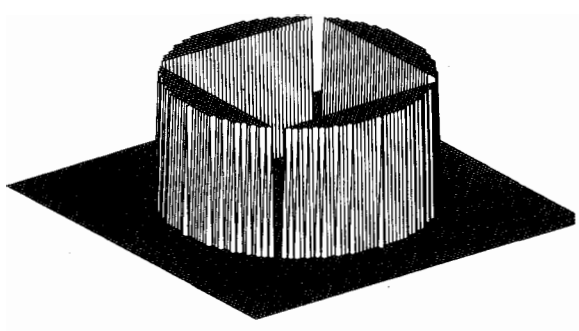

Fig. 14. Circular aperture blocked by an opaque square.

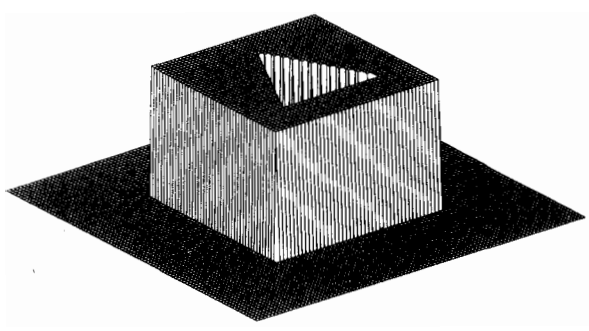

Fig. 16. Square aperture blocked by an opaque triangle.

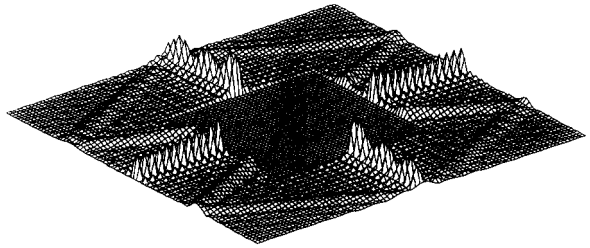

Fig. 17. Blocking low spatial frequencies, Fourier transform modulus portrait.

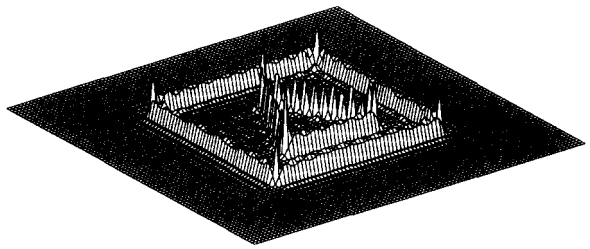

Fig. 18. Edge-detection of the aperture of Fig. 16.

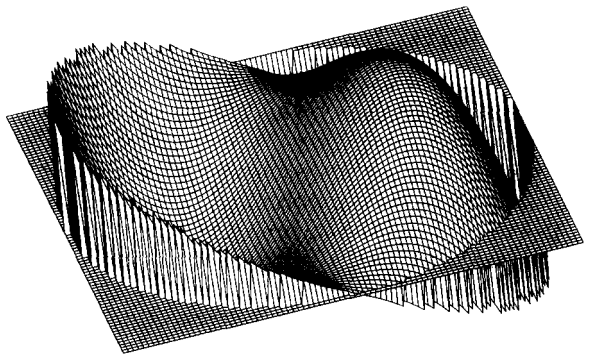

Fig. 19. Phase variation in a coma-aberrated circular aperture.

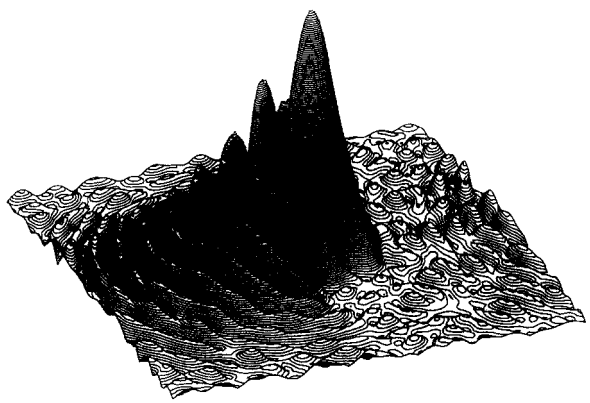

Fig. 20. Modulus of the Fourier transform of the aperture of Fig. 19.

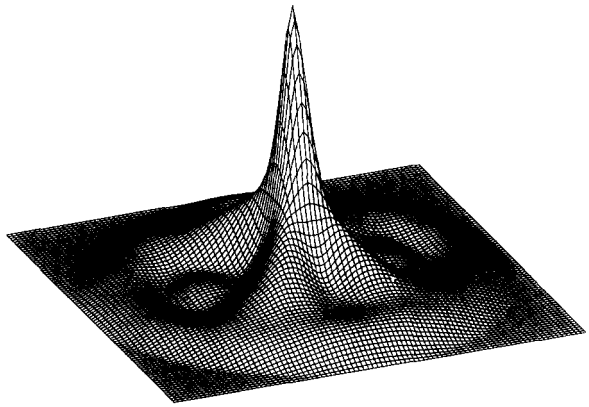

Fig. 21. The MTF of the aperture of Fig. 19.

VI. A COMPLEX APERTURE—COMA

Now into the realm of the more difficult, visualization of a primary aberration, coma, in a circular aperture. This aperture function will be complex. We are guided here by Kim and Shannon.[7] See the Appendix for MATLAB commands. The complex, phase aberrated, round aperture function is shown in Fig. 19, but this is too large to Fourier transform. A smaller aperture is created and the modulus of its Fourier transform is shown in Fig. 20. Even though it is the modulus it is easily recognizable as coma. The point spread function can be created by squaring the (complex) Fourier transform.

The modulation transfer function (MTF) and the phase transfer function (PTF) can be easily created, all commands in the Appendix. Figure 21 is the MTF. All of these figures, especially the phase plots, require some thoughtful interpretation.

VII. FRESNEL DIFFRACTION VIA FOURIER TRANSFORM

The visualization of the results of the coma aberration required the Fourier transformation of a complex function. It can be shown[8] that the Fresnel diffraction amplitude function for a simple aperture can be obtained similarly by Fourier transformation of $\exp(i\pi/\lambda z)(x^2+y^2)$ which must then also be multiplied by two additional complex exponential functions. (λ is the scaled wavelength and z is a scaled distance from the diffracting aperture.) The complex square of this result yields the Fresnel irradiance and as such, the two additional complex functions can be ignored if one is seeking only the irradiance portrayal. Figure 22 is a Fresnel irradiance pattern from a square aperture. We have not examined Fig. 1 to see if convolution could also be a route to Fresnel irradiance, nor have we examined Fresnel diffraction by apertures with specific transmission and/or phase functions.

Glance at the commands for this in the Appendix. Is this not easier than using Cornu spirals and Fresnel integrals?[9] If one wishes to have numerical data, it exists, here in the "fresirr" matrix, and likewise in all the other matrices created as described in this paper.

The command **mesh**(ap) will show the real part of the fictitious aperture used for this calculation. The command **mesh**(ft) will show only the real part of the transform that requires multiplication by two more complex functions in

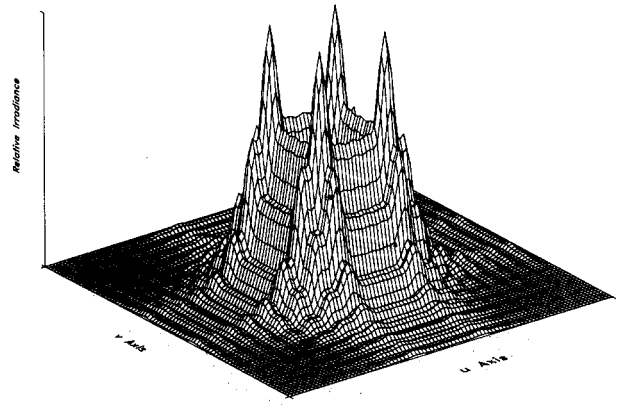

Fig. 22. Fresnel irradiance from a square aperture, obtained by Fourier transformation, see text for scaling.

order to portray the Fresnel amplitude. As the quantity "fres" approaches zero the irradiance pattern calculated, "fresirr," approaches that of Fraunhofer diffraction.

In SURFER the data can be converted for a topographic plot, and it shows the "plaid" appearance often found in Fresnel patterns from rectangular structures. Contour line density is related to irradiance. Thus can be portrayed Fresnel diffraction by *any* simple aperture.

VIII. CONCLUSIONS

The somewhat traditional phasor approach to diffraction and slit interference (perhaps preliminary preparation for Cornu spirals and Fresnel integrals) is quite limited in usefulness. In a general physics course where students know the rudiments of calculus, Fourier spectrum analysis should be an attractive alternative, because a Fourier approach will have *widespread usefulness for students* in many fields of science and engineering. (Reconsider Sec. IV). Reasonably simple apertures Fourier transform with work but with relative ease. We hope we have shown that simulations and visualizations of optical transformations in three-space for any aperture can now be achieved easily on a PC. In an advanced laboratory it would not be too difficult for students to design their own diffraction apertures, for photographic reduction on, say, Kodak Ultratec™ film; and with laser and optical bench create the PSF for comparison with a computer simulated PSF. Even phase steps[10] can be included. Kodak Technical Pan film and a range of exposures might help visualize the detail in the PSFs. See the paper of S. A. Dodds, Ref. 1.

In this paper we have only scratched the surface, so to speak, of computer simulation of the wide ranging techniques of optical processing. There is much fascination to be found in aperture design, matched and complex filtering, optical correlating, optical computing, etc.[11] Computer graphics allows us to visualize more than the eye can see.

APPENDIX—MATLAB COMMANDS

1. Square and circle double aperture:

```
ap = circ(12);
ap(49:72,:) = ones(24);
db = zeros(74,26);
db(2:73,2:25) = ap;
mesh(db)                    aperture display with faulty
                            screen perspective?
otf = conv2(db,db);
mesh(otf)                   displays the OTF. When
                            plotted the perspective was
                            correct. Fig. 12.
```

Here are the cross-(auto-) correlation commands.

sqcrcorr = **xcorr2**(db,db); and then, **mesh**(sqcrcorr).

2. Coma aberrated aperture:

```
[x,y] = meshdom( − 1:2/89:1, − 1:2/89:1);
r = sqrt(x.^2 + y.^2);
theta = a tan 2(y,x);
```

These establish four matrices; all 90×90, one called x, one called y; one called r, one called theta:

```
coma = sqrt(8)*(3*(r.^2) − 2).*r.*cos(theta);
```

see Ref. 7.

This represents the phase in the aperture. The command **mesh**(coma) shows a phase variation in a 90×90 grid. Now put the phase variation into a round aperture:

```
ap = zeros(90);
ap = circ(45).*exp(i*2*pi*(.2)*coma);
mesh(abs(ap))               will show a flat circle,
                            there is only phase
                            variation in the aperture.
mesh(angle(ap))             will display the wave-
                            front (phase)
```

error in the round aperture. The command **unwrap** does not help. We can get rid of some of the spiking (round off error) at the edge of the aperture, by going to an 88×88 array and a 44 radius circle.

```
ap(2:89,2:89)
   = circ(44).*exp(i*2*pi*(.2)*coma(2:89,2:89));
mesh(angle(ap))             should look somewhat better,
                            Fig. 19.
```

This is a good resolution, simple portrayal of a complex aperture but it is too large for Fourier transforming. Make a smaller one:

```
[x,y] = meshdom( − 1:2/19:1, − 1:2/19:1);
r = sqrt(x.^2 + y.^2);
theta = a tan 2(y,x);
coma = sqrt(8)*(3*(r.^2) − 2).*r.*cos(theta);
ap = zeros(90);
ap(36:55,36:55) = circ(10).*exp(i*2*pi*(.2)*coma);
mesh(abs(ap))               will show a flat top cylinder;
mesh(angle(ap))             will show a coma aberrated
```

aperture of appropriate size for Fourier transforming and squaring for the PSF.

Figure 20 is the modulus of the Fourier transform, plotted by SURFER using only lines of constant z.

For the OTF we will use a larger aperture and discrete convolution.

```
[x,y] = meshdom( − 1:2/44:1, − 1:2/44:1);
r = sqrt(x.^2 + y.^2);
theta = a tan 2(y,x);
coma = sqrt(8)*(3*(r.^2) − 2).*r.*cos(theta);
ap = circ(22.5).*exp(i*2*pi*(.2)*coma);
```

```
comaotf = conv2(ap,ap);   whos tells us it is complex.
comamtf = abs(comaotf);   creates the modulus, the MTF, Fig. 21.
comaptf = angle(comaotf); creates PTF
```

3. Fresnel irradiance by Fourier transform. Here is a procedure for a Fresnel irradiance portrait of a simple square aperture where $\lambda z = 1/2$:

```
[x,y] = meshdom( − 1:2/19:1, − 1:2/19:1);
fres = (x.^2 + y.^2)*(1);
ap = zeros(90);
ap(37:56,37:56) = ones(20).*exp(i*2*pi*fres);
ft = fftshift(fft2(fftshift(ap)));
fresirr = ft.*conj(ft);
mesh(fresirr)          See Fig. 22.
```

4. Triangle (not discussed above). An equilateral triangle (with two ragged edges) of appropriate size for transformation can be created:

```
ap = zeros(89);
for i = 35:45
jmin = (6/10)*(i − 45) + 45;
jmax = − (6/10)*(i − 45) + 45;
for j = jmin:jmax
ap(i, j) = 1;
end;
end;
mesh(ap)          yields the aperture matrix.
ft = fftshift(fft2(fftshift(ap)));   for the Fourier transform.
```

MATLAB returns the real part of the transform on **mesh**(ft); or use reft = **real**(ft); and **mesh**(reft). **abs**(ft) returns the modulus, **imag**(ft) returns the imaginary part. The command **angle**(ft) and a few unwraps will return a phase plot of the transform.

For the OTF a larger triangle can be established:

```
ap = zeros(40);
for i = 1:39
jmin = (1/sqrt(3))*(i − 39) + 23;
jmax = − (1/sqrt(3))*(i − 39) + 23;
for j = jmin: jmax
ap(i, j) = 1;
end;
end;
mesh(ap)          yields the larger aperture.
triotf = conv2(ap,ap);
mesh(triotf)      yields the OTF.
```

This last calculation seems slow even on a 386 computer. Smith and Marsh[12] have shown how to do some of these calculations by hand.

ACKNOWLEDGMENTS

This work had its origin via NSF Grant SED-8021473. David Botkin made programming contributions in the early stages. Patra Noonan prepared the manuscript. Illinois Wesleyan University greatly facilitated the work by providing contemporary computing hardware and software.

[1]For example: C. A. Bennett, "A computer-assisted experiment in single-slit diffraction and spatial filtering," Am. J. Phys. **58**, 75–78 (1990); S. A. Dodds, "An optical diffraction experiment for the advanced laboratory," Am. J. Phys. **58**, 663–668 (1990); X. Chen, J. Huang, and E. Loh, "Two-dimensional fast Fourier transform and pattern processing with IBM PC," Am. J. Phys. **56**, 747–749 (1988).

[2]SURFER™, Golden Software Inc., P.O. Box 281, Golden, CO 80402.

[3]MATLAB™, The Math Works, 21 Eliot St., South Natick, MA 01760.

[4]R. J. Higgins, "Fast Fourier transform: An introduction with some minicomputer experiments," Am. J. Phys. **44**, 766–773 (1976); P. Maas, "Discussion of the use of fast Fourier transform for spectral analysis," Am. J. Phys. **46**, 857–858 (1978).

[5]Draft International Standard ISO/DIS 9334, 1987.

[6]D. R. Kaiser and B. R. Russell, "Demonstration of the phase reversal effect in Fraunhofer diffraction patterns," Am. J. Phys. **48**, 674–675 (1980).

[7]C-J. Kim and R. R. Shannon, "Catalog of Zernike polynomials," in *Applied Optics and Engineering, Vol. X* (Academic, San Diego, CA, 1987), Chap. 4, pp. 193–221. In modern optics textbooks consult indexes for "wave aberrations."

[8]G. O. Reynolds, J. B. DeVelis, G. B. Parrent, Jr., and B. J. Thompson, *The New Physical Optics Notebook: Tutorials in Fourier Optics* (SPIE-The International Society for Optical Engineering, and American Institute of Physics, Bellingham, WA, 1989), pp. 59–60; J. W. Goodman, *Introduction to Fourier Optics* (McGraw-Hill, New York, 1968), p. 60; "Operations achievable with coherent optical information processing systems," Proc. IEEE **65**, 29–38 (1977); K. Iizuka, *Engineering Optics—2nd Ed.* (Springer-Verlag, New York, 1987), pp. 62–68.

[9]M. A. Heald, "Computation of Fresnel diffraction," Am. J. Phys. **54**, 980–983 (1986).

[10]H. R. Stark, W. R. Bennett, and M. Arm, "Design considerations in power spectra measurements by diffraction of coherent light," Appl. Opt. **8**, 2165–2172 (1969).

[11]J. L. Horner and P. D. Gianino, "Phase-only matched filtering," Appl. Opt. **23**, 812–816 (1984); D. McLachlan, Jr., "The role of optics in applying correlation functions to pattern recognition," J. Opt. Soc. Am. **52**, 454–459 (1962); F. T. S. Yu and E. Y. Wang, "Undergraduate coherent optics laboratory," Am. J. Phys. **41**, 1160–1169 (1973); J. C. Brown, "Fourier analysis and spatial filtering," Am. J. Phys. **39**, 797–801 (1971); J. P. Mills and B. J. Thompson, "Effect of aberrations and apodization on the performance of coherent optical systems. I. The amplitude impulse response," J. Opt. Soc. Am. A **3**, 694–703 (1986); A. Hairie and J. Provost, "Optical correlator using the incoherent light of a slide projector," Am. J. Phys. **51**, 832–836 (1983); A. B. Meinel, M. P. Meinel, and N. J. Woolf, "Multiple aperture telescope diffraction images," in *Applied Optics and Optical Engineering, Vol. IX* (Academic, New York, 1983), Chap. 5, pp. 149–201.

[12]R. C. Smith and J. S. Marsh, "Diffraction patterns of simple apertures," J. Opt. Soc. Am. **64**, 798–803 (1974).