```
// power of 2 for optimum speed
int N = 512;
// distance from aperture to screen
double z = 0.5e+6;
FFT2D fft2d = new FFT2D(N, N);
double[] cdata = new double[2*N*N]; // complex data
double a = 6000;                    // aperture mask dimension
double dx = 2*a/N, dy = 2*a/N;
double x = -a;
for(int ix = 0;ix<N;ix++) {
   int offset = 2*ix*N;
   double y = -a;
   for(int iy = 0;iy<N;iy++) {
      double r2 = (x*x+y*y);
      cdata[offset+2*iy] = (r2<4e6) ? 1 : 0; // circular aperture
      cdata[offset+2*iy+1] = 0;
      y += dy;
   }
   x += dx;
}
fft2d.transform(cdata);
// get arrays containing the wavenumbers in wrapped order
double[] kx = fft2d.getWrappedOmegaX(-a, a);
double[] ky = fft2d.getWrappedOmegaY(-a, a);
for(int ix = 0;ix<N;ix++) {
   int offset = 2*ix*N; //offset to beginning of row
   for(int iy = 0;iy<N;iy++) {
      double radical = PI4-kx[ix]*kx[ix]-ky[iy]*ky[iy];
      if(radical>0) {
         double phase = z*Math.sqrt(radical);
         double real = Math.cos(phase);
         double imag = Math.sin(phase);
         double temp = cdata[offset+2*iy];
         cdata[offset+2*iy] =
              real*cdata[offset+2*iy]-imag*cdata[offset+2*iy+1];
         cdata[offset+2*iy+1] =
              real*cdata[offset+2*iy+1]+imag*temp;
      } else { //evanescent waves decay exponentially
         double decay = Math.exp(-z*Math.sqrt(-radical));
         cdata[offset+2*iy] *= decay;
         cdata[offset+2*iy+1] *= decay;
      }
   }
}
fft2d.inverse(cdata);
double max = 0;
for(int i = 0;i<N*N;i++) {      // find max intensity
   double real = cdata[2*i];    // real
   double imag = cdata[2*i+1]; // imaginary
   max = Math.max(max, real*real+imag*imag);
}
// intensity is squared magnitude of the amplitude
int[] data = new int[N*N];
for(int i = 0, N2 = N*N;i<N2;i++) {
   double real = cdata[2*i];
   double imag = cdata[2*i+1];
```


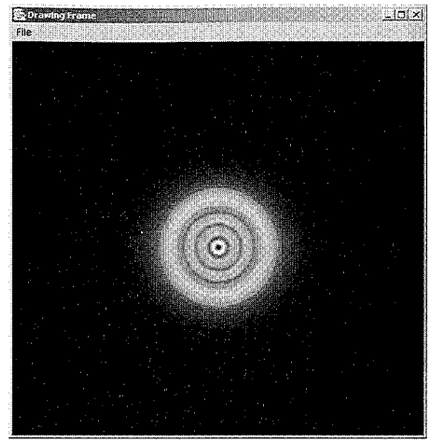

Figure 9.5 Computed Fresnel diffraction on a screen illuminated by a uniform plane wave and located $0.5 \times 10^6\lambda$ from a circular aperture of radius 2000λ.

```
   data[i] = (int) (255*(real*real+imag*imag)/max);
}
// raster for least memory and best speed
RasterFrame frame = new RasterFrame("Fraunhofer Diffraction");
frame.setBWPalette();
frame.setAll(data, N, -0.5, 0.5, -0.5, 0.5);
frame.setVisible(true);
frame.setDefaultCloseOperation(javax.swing.JFrame.EXIT_ON_CLOSE);
  }
}
```

Because the algorithm in Listing 9.12 depends only on the linearity of the wave equation, it is exact and may be applied to many practical optics problems. Its main limitation occurs when z_0 is large because of rapid oscillations in the phase factor. In this case, the far field (Fraunhofer) approximation usually becomes applicable and should be used.

Problem 9.39 Fresnel diffraction

The diffraction pattern from a circular aperture is important because most lenses, mirrors, and optical instruments have cylindrical symmetry. (See Figure 9.5.)

(a) Compute the diffraction pattern due to a circular aperture with a radius of 1000λ at a screen distance of $10^5\lambda$. Is the center of the diffraction pattern dark or light? Reposition the screen to $2 \times 10^5\lambda$ and repeat the calculation. Does the intensity at the center of the shadow change? Use a 512×512 grid to sample a region of space 5000λ on a side.

(b) Replace the screen by a circular disk. That is, use an aperture mask that is opaque if $r < 1000\lambda$. Is the center of the screen light or dark? Does the center change from bright to dark if the screen is repositioned? ■