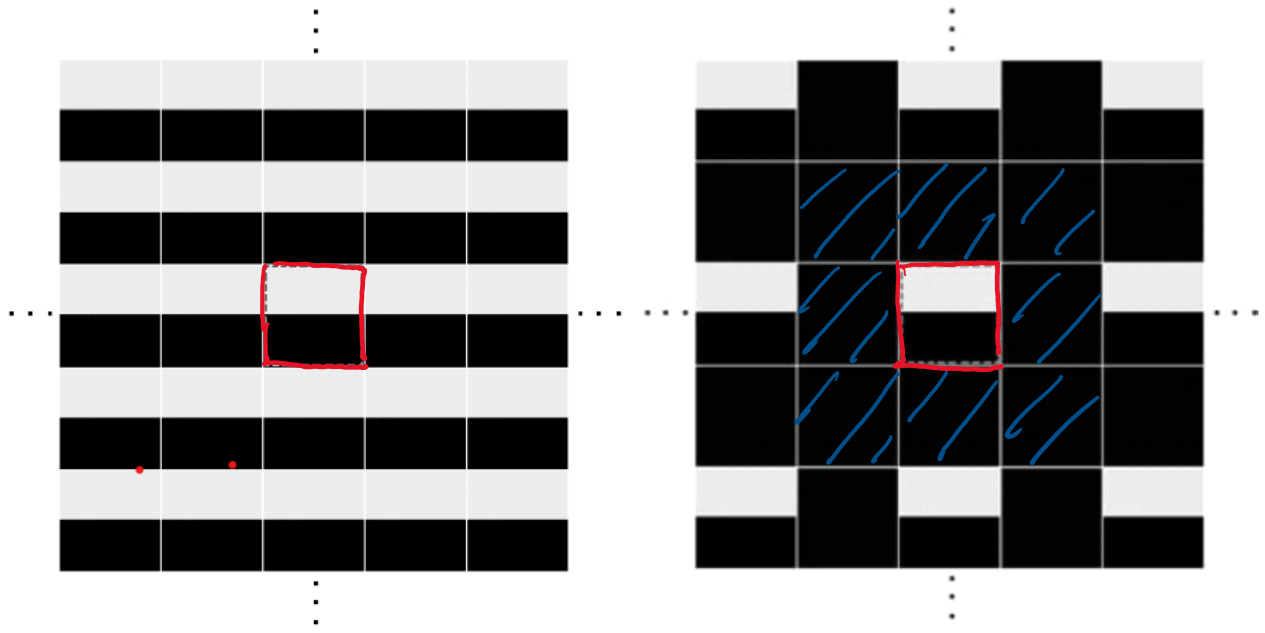
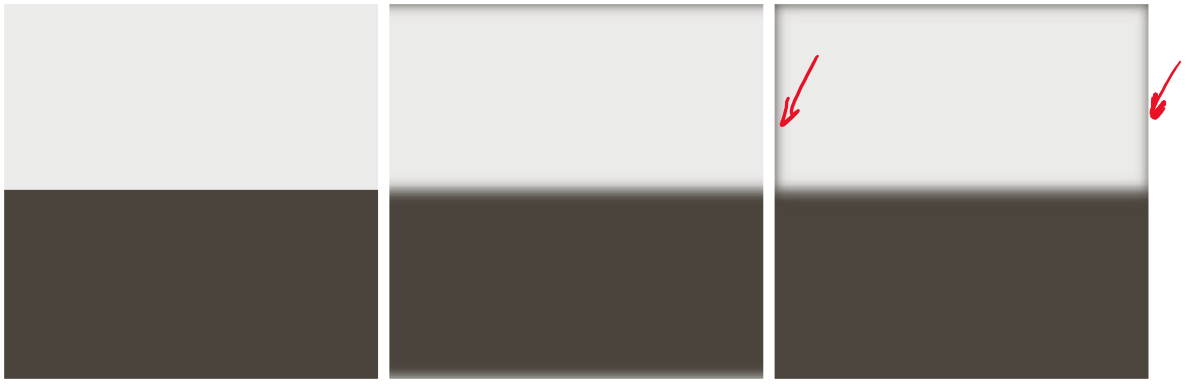
Lecture 25

Wraparound Errors due to periodicity implied by DFT



a b

FIGURE 4.33 2-D image periodicity inherent in using the DFT. (a) Periodicity without image padding. (b) Periodicity after padding with 0s (black). The dashed areas in the center correspond to the image in Fig. 4.32(a). (The thin white lines in both images are superimposed for clarity; they are not part of the data.)



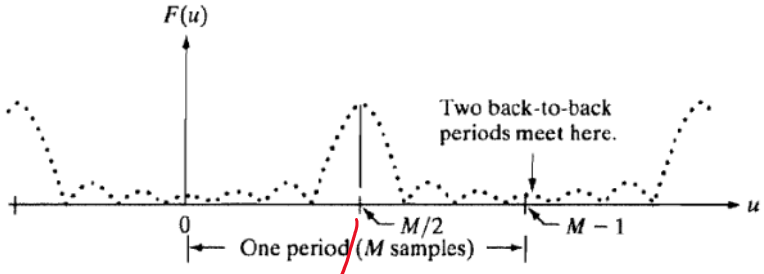
a b c

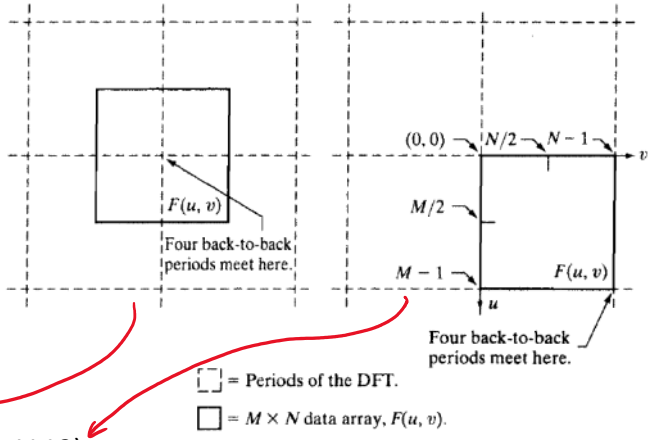
FIGURE 4.32 (a) A simple image. (b) Result of blurring with a Gaussian lowpass filter without padding. (c) Result of lowpass filtering with padding. Compare the light area of the vertical edges in (b) and (c).

Un-centered Spectrum vs. Centered Spectrum

http://www.ece.uah.edu/~dwpan/course/ee604/code/ch4/dftuv.m

```
function [U, V] = dftuv(M, N)
%DFTUV Computes meshgrid frequency matrices.
%   [U, V] = DFTUV(M, N) computes meshgrid frequency matrices U and
%   V.  U and V are useful for computing frequency-domain filter
%   functions that can be used with DFTFILT.  U and V are both
%   M-by-N.

% Set up range of variables.
u = 0:(M - 1);
v = 0:(N - 1);
% Compute the indices for use in meshgrid.
idx = find(u > M/2);
u(idx) = u(idx) - M;
idy = find(v > N/2);
v(idy) = v(idy) - N;
% Compute the meshgrid arrays.
[V, U] = meshgrid(v, u);
```






```
>> [U, V] = dftuv(5, 5)

U =
     0     0     0     0     0
     1     1     1     1     1
     2     2     2     2     2
    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1

V =
     0     1     2    -2    -1
     0     1     2    -2    -1
     0     1     2    -2    -1
     0     1     2    -2    -1
     0     1     2    -2    -1

>> U.^2 + V.^2
ans =
     0     1     4     4     1
     1     2     5     5     2
     4     5     8     8     5
     4     5     8     8     5
     1     2     5     5     2

>> fftshift(U.^2 + V.^2)
ans =
     8     5     4     5     8
     5     2     1     2     5
     4     1     0     1     4
     5     2     1     2     5
     8     5     4     5     8
```

Center $\left(\frac{M}{2}, \frac{N}{2}\right)$

http://www.ece.uah.edu/~dwpan/course/ee604/code/ch4/lpfilter.m

```
function H = lpfilter(type, M, N, D0, n)
%LPFILTER Computes frequency domain lowpass filters
%   H = LPFILTER(TYPE, M, N, D0, n) creates the transfer function of
%   a lowpass filter, H, of the specified TYPE and size (M-by-N).  To
%   view the filter as an image or mesh plot, it should be centered
%   using H = fftshift(H).

%  'gaussian' Gaussian lowpass filter with cutoff (standard deviation)
%             D0.  n need not be supplied.  D0 must be positive.

% Use function dftuv to set up the meshgrid arrays needed for
% computing the required distances.
[U, V] = dftuv(M, N);
% Compute the distances D(U, V).
D = sqrt(U.^2 + V.^2);
```

In Comparison,

```
function [H] = lpfilter_center(type,M,N,D0,n)
% LPFILTER Computes freq. domain lowpass filters

L1 = M/2;
L2 = N/2;

[U,V] = meshgrid(0:M-1,0:N-1);
%   Compute the filter center
U0=L1;
V0=L2;
%             Compute distances D(U,V)
D=((U-U0).^2 + (V-V0).^2).^0.5;
%             Begin filter computations
```

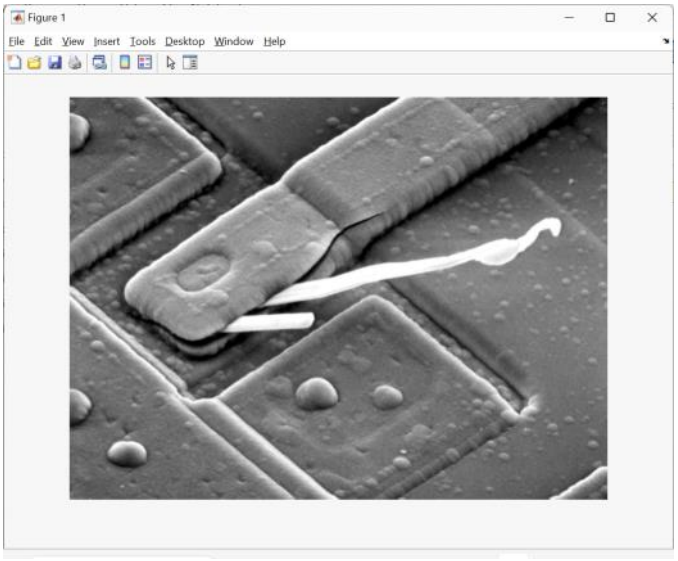
$$D = \sqrt{\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2}$$

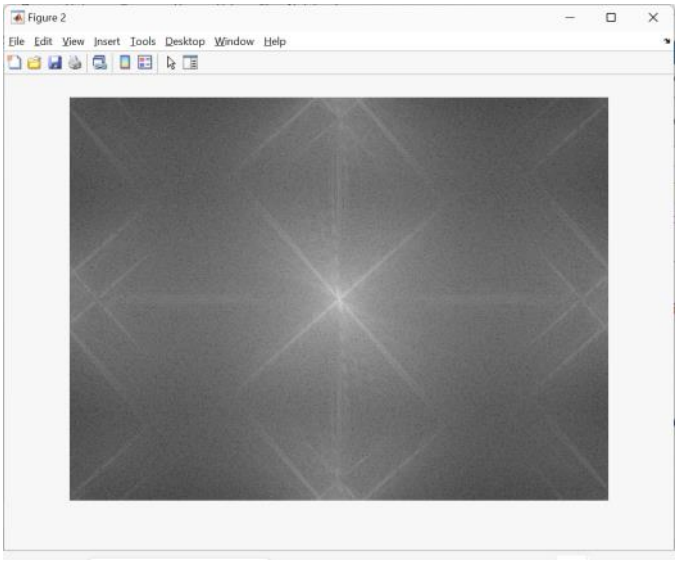
```
>> [U,V] = meshgrid(0:5-1,0:5-1)

U =
     0     1     2     3     4
     0     1     2     3     4
     0     1     2     3     4
     0     1     2     3     4
     0     1     2     3     4

V =
     0     0     0     0     0
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4

>> D=((U-5/2).^2 + (V-5/2).^2).^0.5
D =
    3.5355    2.9155    2.5495    2.5495    2.9155
    2.9155    2.1213    1.5811    1.5811    2.1213
    2.5495    1.5811    0.7071    0.7071    1.5811
    2.5495    1.5811    0.7071    0.7071    1.5811
    2.9155    2.1213    1.5811    1.5811    2.1213
```

Another filtering example:

```
>> I = imread('Fig0429(a)(blown_ic).tif');
>> imshow(I)
```

```
>> F = fft2(I);
>> S = fftshift((log(1 + abs(F))));
>> figure; imshow(S, [])
```

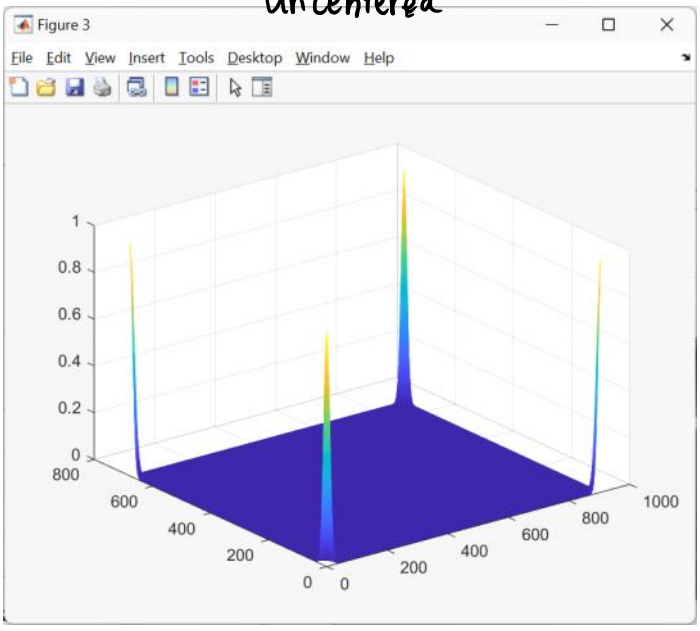




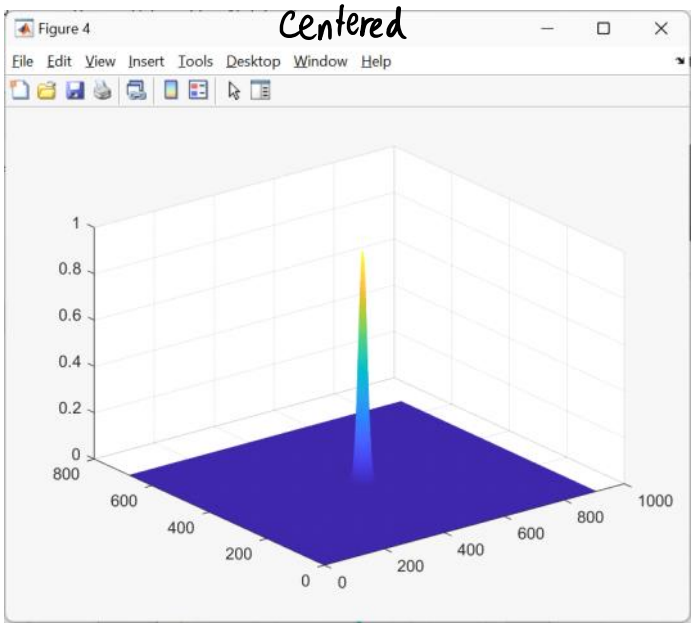

```
>> H = lpfilter('gaussian', M, N, 10);
>> figure; mesh(H)
```

```
>> figure; mesh(fftshift(H))
```

Uncentered

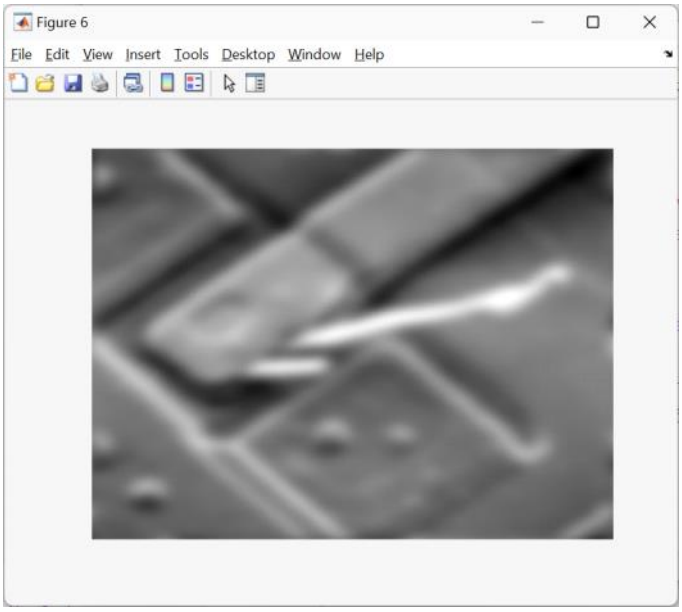


Centered



```
% Fig. 4.36
% Filtering in frequency domain
...
F = fft2(f);
sigma = 10;
H = lpfilter('gaussian', M, N, sigma);
G = F .* H;
g = real(ifft2(G));
imshow(g, [], 'initialmagnification','fit');
```

% uncentered



-- Spatial vs. Frequency Domain Filtering

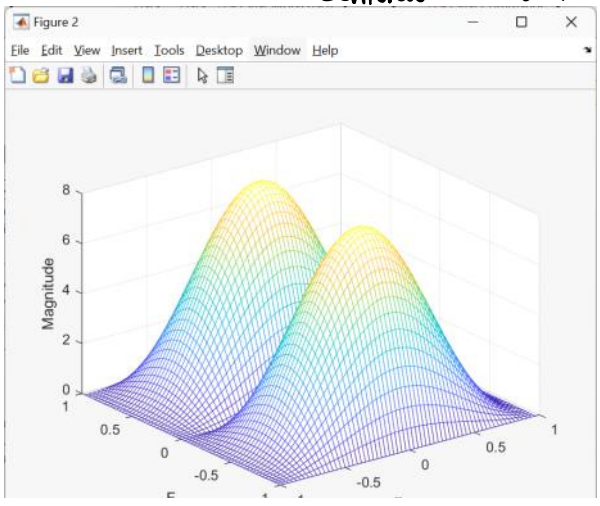
```
>> I = imread('Fig0438(a)(bld_600by600).tif');

>> h = fspecial('sobel')
h =
    1    2    1
    0    0    0
   -1   -2   -1
```

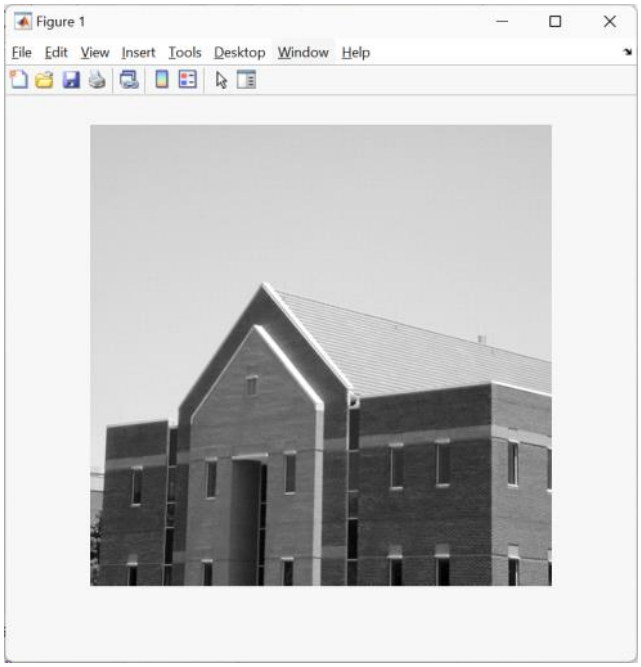
Freqz2

2-D frequency response

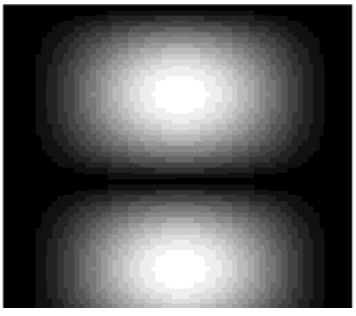
```
>> figure; freqz2(h)
```

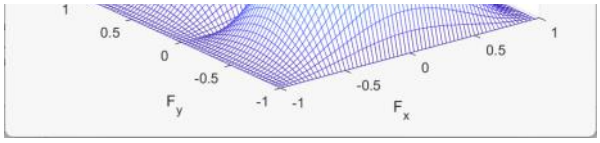
centered transfer function



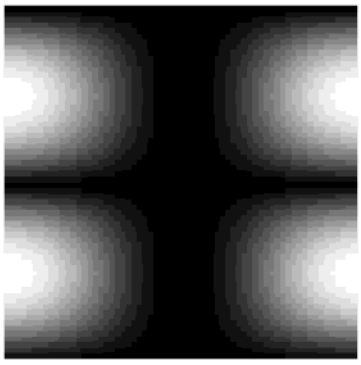



```
>> H = freqz2(h);
>> figure; imshow(abs(H), [])
```




1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1
F_y
F_x

>> figure; imshow(ifftshift(abs(H)), [])



← uncentered

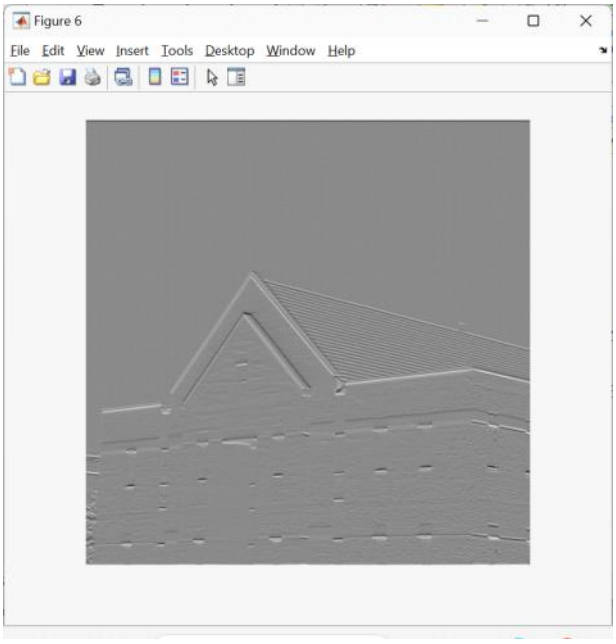
Zero-padding the transfer function and the input image

```
>> PQ = paddedsize(size(I))
PQ =
      1200       1200
>> H = freqz2(h, PQ(1), PQ(2));
>> whos H
  Name        Size              Bytes  Class     Attributes
  H        1200x1200          23040000  double    complex
```

```
>> H1 = ifftshift(H);
```

http://www.ece.uah.edu/~dwpan/course/ee604/code/ch4/dftfilt.m

```
function g=dftfilt(f,H)
%DFTFILT Performs frequency domain filtering.
%Obtain the FFT of the padded input.
F=fft2(f, size(H,1),size(H,2));
%Perform filtering.
g=real(ifft2(H.*F));
%Crop to original size.
g=g(1:size(f,1),1:size(f,2));
```



```
>> gf = dftfilt(I, H1);
>> figure; imshow(gf, [])
```