EIE4512 - Digital Image Processing

Week 4 Tutorial

Bin Zeng

zengbin@cuhk.edu.cn
School of Science and Engineering
The Chinese University of Hong Kong, Shen Zhen

February 21, 2019

Content

1. Function Reconstruction (Recovery) from Sampled Data
2. Computing and Visualizing the 2-D DFT
3. Lowpass Frequency Domain Filters
4. Sharpening Frequency Domain Filters
5. Practise Time

1. Function Reconstruction (Recovery) from Sampled Data

- reconstruction of a function from a set of its samples reduces in practice to interpolating between the samples

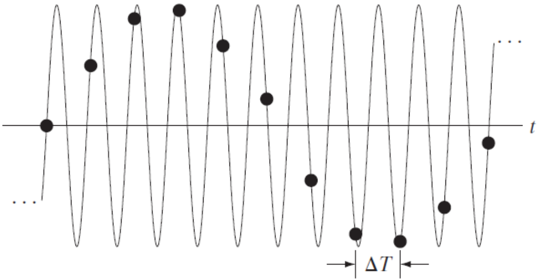
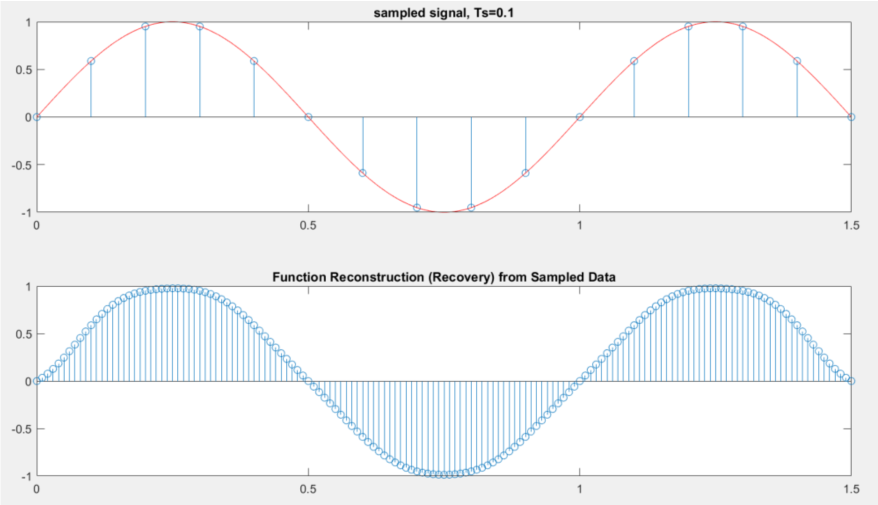


FIGURE 4.10 Illustration of aliasing. The under-sampled function (black dots) looks like a sine wave having a frequency much lower than the frequency of the continuous signal. The period of the sine wave is 2 s, so the zero crossings of the horizontal axis occur every second. ΔT is the separation between samples.

1. Function Reconstruction (Recovery) from Sampled Data

$$
\begin{aligned}
f(t) &= FT^{-1}\left[\tilde{F}(\mu)H(\mu)\right] \\
&= \sum_{n=-\infty}^{\infty} f(n\Delta T)\cdot \operatorname{sinc}\left[\frac{1}{\Delta T}(t-n\Delta T)\right]
\end{aligned}
$$

1. Function Reconstruction (Recovery) from Sampled Data



2. Computing and Visualizing the 2-D DFT

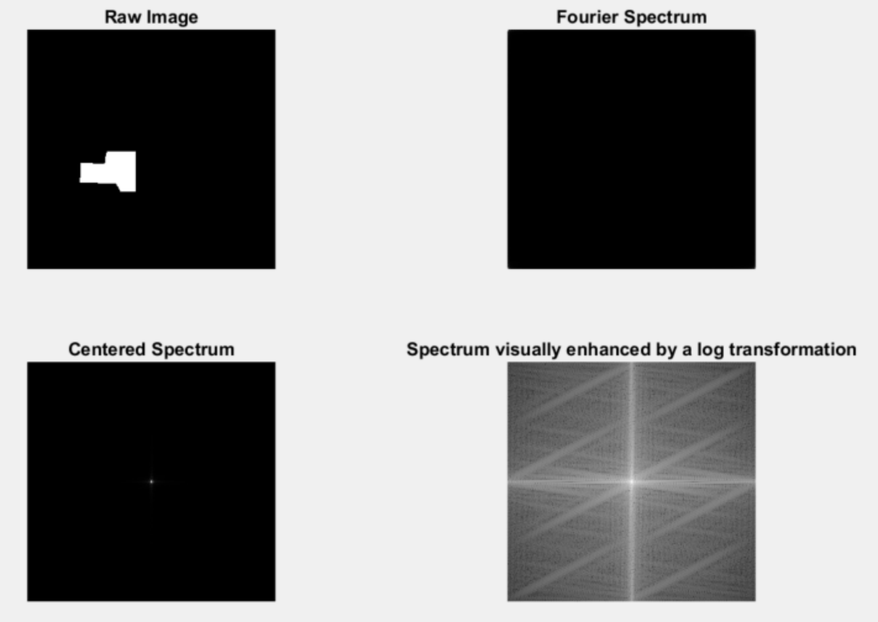
2-D *Discrete Fourier Transform* (DFT) of a digital image *f(x,y)* of size *M*x*N*:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+vy/N)}$$

Inverse Discrete Fourier Transform (IDFT):

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux/M+vy/N)}$$

2. Computing and Visualizing the 2-D DFT

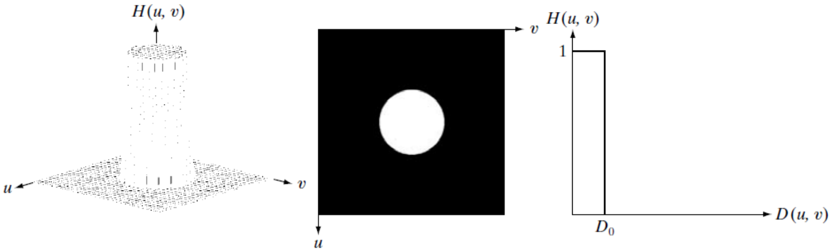


3. Lowpass Frequency Domain Filters

- A 2-D lowpass filter that passes without attenuation all frequencies within a circle of radius D0 from the origin and cuts off all frequencies outside this circle is called an ideal lowpass filter (ILPF);

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$$

3. Lowpass Frequency Domain Filters



a b c

FIGURE 4.40 (a) Perspective plot of an ideal lowpass-filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

3. Lowpass Frequency Domain Filters

- Matlab Implement

```
% 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);

% Begin filter computations.
switch type
case 'ideal'
   H = double(D <= D0);
case 'btw'
   if nargin == 4
      n = 1;
   end
   H = 1./(1 + (D./D0).^(2*n));
case 'gaussian'
   H = exp(-(D.^2)./(2*(D0^2)));
otherwise
   error('Unknown filter type.')
```

4. Sharpening Frequency Domain Filters

- Just as lowpass filtering blurs an image, the opposite process, highpass filtering, sharpens the image by attenuating the low frequencies and leaving the high frequencies of the Fourier transform relatively unchanged.

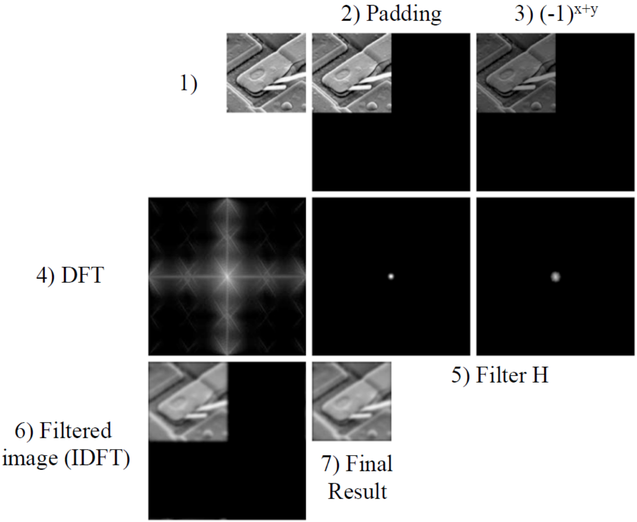
```
function H = hpfilter(type, M, N, D0, n)

if nargin == 4
   n = 1; % Default value of n.
end

% Generate highpass filter.
Hlp = lpfilter(type, M, N, D0, n);
H = 1 - Hlp;
```

5. Practise Time

- Implement frequency domain filtering step by step



a b c
d e f
g h

FIGURE 4.36 (a) An $M \times N$ image, f. (b) Padded image, f_p of size $P \times Q$. (c) Result of multiplying f_p by $(-1)^{x+y}$. (d) Spectrum of F_p. (e) Centered Gaussian lowpass filter, H, of size $P \times Q$. (f) Spectrum of the product HF_p. (g) g_p, the product of $(-1)^{x+y}$ and the real part of the IDFT of HF_p. (h) Final result, g, obtained by cropping the first M rows and N columns of g_p.