# Computer

## Image Processing II

# Recap

- Spatial domain, intensity transformations (on single pixels)
  - Image thresholding
    - Otsu's method
    -
    -
  - Image negati
  - Log transform
  - Power-law
  - Piecewise-linear transformation
    - Contrast stretching
    - Gray-level slicing
    - Bit-plane slicing
  - Histogram processing
    - Histogram equalization
    - Histogram matching

# Recap

- Spatial domain, intensity transformations (on single pixels)
  - Histogram processing
    - Histogram equalization
    - Histogram matchi
  - Arithmetic/Logic O
    - +, -, AND, OR, XOR
    - Image averaging
- Spatial Filtering (using neighbouring pixe
    - Smoothing
    - Gaussian Filter
    - Median Filter
    - Pooling
    - Laplacian
    - Padding

# Frequency Domain Techniques

**Goal**:



- to gain working knowledge of Fourier transform and frequency do cessing.





- focus on fundamentals and re mage Processing

- **not** signal processing expertise!

# Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

> ***Any*** *univariate function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's (mostly) true!
  - called Fourier Series
  - there are some subtle restrictions

*...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.*

**Legendre**

# A sum of sines

Our building block:



$$A\sin(\omega x + \phi)$$



Add enough of them to get any signal *g(x)* you want!

# Frequency VS Spatial Domain

- Spatial domain
  - the image plane itself
  - direct manipul
  - changes in pixe ges in the scene
- Frequency domain
  - Fourier transform of an image
  - directly related to rate of changes in the image
  - changes in pixel position correspond to changes in the spatial frequency

# Frequency Domain Overview

- Frequency in image



  - high frequencies correspond to pixel values that change rapidly across the ima



  - low frequency rge scale features in the image



- Frequency domain
  - defined by values of the Fourier transform and its frequency variables (u, v)

# Frequency Domain Overview

Frequency domain processing

# Fourier Series

- Periodic function can be represented as a weighted sum of sines and cosines of different frequencies
- Even functions that are not periodic (but whose area under the curve is finite) can be ex                                        s and/or cosines multiplied by a we



sum =

# Sum of Sines

# 1-D Fourier Transform and its Inverse

For a single variable continuous function f(x), the Fourier Transform F(u) is defined by:



$$\qquad \infty \qquad (\quad) \quad ^{-j2\pi ux} dx \qquad (1)$$

where: $j = \sqrt{-1}$



Given F(u), we recover f(x) using the I Transform:



$$\mathrm{f(x)} = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du \qquad (2)$$

(1) and (2) constitute a Fourier transform pair

# 2-D Fourier Transform and its Inverse

In two dimensions, we have:



$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\, e^{-j2\pi(ux+vy)}\, dx\, dy \qquad (3)$$





$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)\, e^{j2\pi(ux+vy)}\, du\, dv \qquad (4)$$

# Discrete Fourier Transform

- In one dimension,



$$\mathrm{F(u)} = \frac{1}{M}\sum_{x=}^{M-} (\quad) \quad ^{-\frac{j2\pi ux}{}} \qquad ,1,2,\ldots,M\text{-}1 \qquad (5)$$

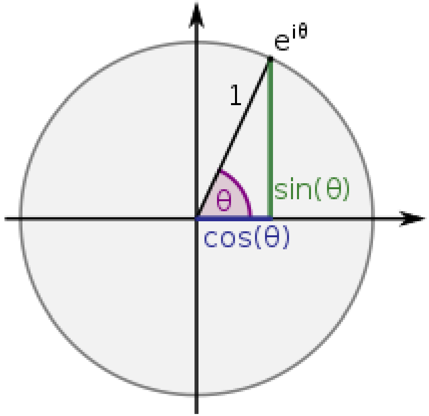


$$\mathrm{f(x)} = \sum_{u=0}^{M-1} F(u)\, e^{\frac{j2\pi ux}{M}} \qquad 2,\ldots,M\text{-}1 \qquad (6)$$



- Note that the location of $1/\mathrm{M}$ does not matter, so long as the product of the two multipliers is $1/\mathrm{M}$
- Also in the discrete case, the Fourier transform and its inverse always exist

# Discrete Fourier Transform




Consider Euler's formula:

$$e^{j\theta} = \cos(\theta) + j\sin(\theta) \quad (7)$$



Substituting this expre

$$\cos(-\theta) = \cos(\theta),$$



$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)\left(\cos\frac{2\pi ux}{M} - j\sin\frac{\quad}{}\right) \quad = 0,1,2, \ldots, \text{M-1} \quad (8)$$



- Each term of F depends on all values of f(x), and values of f(x) are multiplied by sines and cosines of different frequencies.
- The domain over which values of F(u) range is called the *frequency domain*, as *u* determines the frequency of the components of the transform.

# 2-D Discrete Fourier Transform

Digital images are 2-D discrete functions:



$$F(u,v) \quad \frac{1}{\quad} \quad \sum^{M-1} \sum^{N} \quad ( \quad ) \quad {}^{-j2\pi\left(\frac{ux}{\quad}+\frac{vy}{N}\right)}$$



$$for\ u = 0,1,2, \quad \quad = 0,1,2,\ldots,N-1. \qquad (9)$$



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

$$for\ x = 0,1,2,\ldots,M-1\ and\ y = 0,1,2,\ldots,N-1. \qquad (10)$$

# Frequency Domain Filtering

- Frequency is directly related to **rate of change**, so frequencies in the Fourier transform may be related to **patterns of intensity variations in the image**.
- Slowest varying fr [illegible] nds to average gray level of the image.
- Low frequencies correspond to slow [illegible] ponents in the image- for example, large areas of similar gray levels.
- Higher frequencies correspond to faster gray level changes- such as edges, noise etc.

# Procedure for Filtering in the Frequency Domain

1. Multiply the input image by $(-1)^{x+y}$ to centre the transform at (M/2, N/2), which is the centre of the MxN area occupied by the 2D DFT
2. Compute the DF
3. Multiply $F(u,v)$ b
4. Compute the inverse DFT transfor
5. Obtain the real part $g(x,y)$
6. Multiply the result by $(-1)^{x+y}$

# Example: Notch Filter

- We wish to force the average value of an image to zero. We can achieve this by setting $F(0, 0) = 0$, and then taking its inverse transform.



- So choose the filter function as:

$$\begin{cases} H(u, v) = 0 \text{ if } (u, v) = (M/2, N/2) \\ H(u, v) = 1 \text{ otherwise.} \end{cases}$$



- Called the **notch filter**- constant functio [illegible] otch) at the origin.



- A filter that attenuates high frequencies while allowing low frequencies to pass through is called a lowpass filter.
- A filter that attenuates low frequencies while allowing high frequencies to pass through is called a highpass filter

# Convolution Theorem: correspondence between spatial and frequency domain filtering

Let F(u, v) and H( u, v) be the Fourier transforms of f(x, y) and h(x,y). Let * be spatial convolution, and multiplication be element-by-element product. Then



- f(x, y) * h(x, y) an ... ourier transform pair



- Analogously, conv ... ain reduces to multiplication in the spatial domain, ... a.



$$(f * h)(t) \Leftrightarrow (H \cdot F)(u) \qquad (f \cdot h)(t) \Leftrightarrow (H * F)(u)$$

Using this theorem, we can also show that filters in the spatial and frequency domains constitute a Fourier transform pair.

# Exploiting the correspondence

- If filters in the spatial and frequency domains are of the same size, then filtering is more efficient computationally in frequency domain.



- However, spatial f e.



- Filtering is also mo ain- so design it there.



- Then, take the inverse transform, an ulting filter as a guide to design smaller filters in the spatial domain.

# Example of Smoothing an Image

- In spatial domain, we just convolve the image with a Gaussian kernel to smooth it
- In frequency do image by a filter achieve the sam

# Example of Smoothing an Image

1. Multiply the input image by $(-1)^{x+y}$ to center the transform
2. Compute the DFT F(u,v) of the resulting image
3. Multiply F(u,v)
4. Computer the inverse DFT tran )
5. Obtain the real part h(x,y) of 4
6. Multiply the result by $(-1)^{x+y}$

# Example of Smoothing an Image

1. Multiply the input image by $(-1)^{x+y}$ to center the transform
2. Compute the DFT F(u,v) of the resulting image
3. Multiply F(u,v)
4. Computer the inverse DFT tran )
5. Obtain the real part h(x,y) of
6. Multiply the result by (-1)

# Example of Smoothing an Image

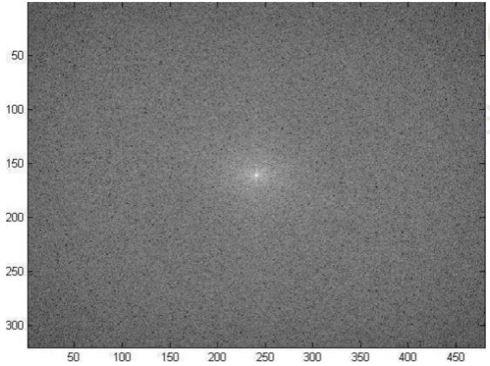
1.Multiply the input image by $(-1)^{x+y}$ to center the transform

2.Compute the DFT F(u,v) of the resulting image



3.Multiply F(u,v)





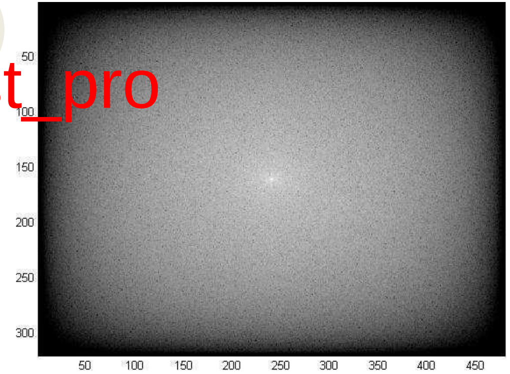

e inverse DFT tran )



eal part h(x,y) of 4

X

result by $(-1)^{x+y}$



# Example of Smoothing an Image

1. Multiply the input image by $(-1)^{x+y}$ to center the transform
2. Compute the DFT F(u,v) of the resulting image
3. Multiply F(u,v) by a filter G(u,v)

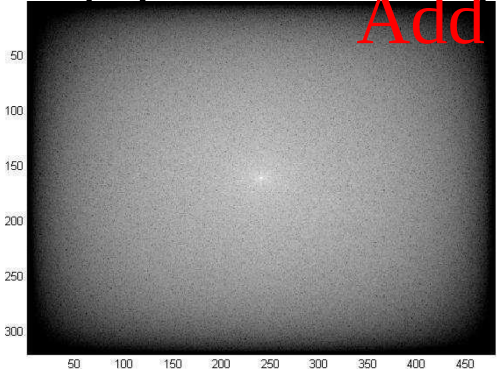


4. Compute the inverse DFT transform h*(x,y)
5. Obtain the real



6. Multiply the result by $(-1)^{x+y}$





# Gaussian Filter

- Gaussian filters are important because their shapes are easy to specify, and both the forward and inverse Fourier transforms of a Gaussian function are real Gaussian functions.



- Let H(u) be a one di d by:



$$H(u) = A \exp^{-\frac{u^2}{2\sigma^2}}$$

- where σ is the standard deviation of t rve.



- The corresponding filter in the spatial domain is

$$h(x) = \sqrt{2\pi\sigma}\, A \exp^{-2\pi^2\sigma^2 x^2}$$

- This is usually a lowpass filter.

# Difference of Gaussian - DoG Filter

- Difference of Gaussians may be used to construct highpass filters:



$$\mathrm{H(u)} = \mathrm{A}\exp^{-\frac{u^2}{2\sigma_1^2}} - \mathrm{B}\exp^{-\frac{u^2}{2\sigma_2^2}}$$



with $A \geq B$ and $\sigma_1 > \sigma_2$.



- The corresponding filter in the spatial

$$\mathrm{h(x)} = \sqrt{2\pi\sigma_1}\,\mathrm{A}\exp^{-2\pi^2\sigma_1^2x^2} - \sqrt{2\pi\sigma_2}\,\mathrm{B}\exp^{-2\pi^2\sigma_2^2x^2}$$

# Why does a lower resolution image still make sense to us? What do we lose?

# Multiresolution Processing

- Small objects, low contrast benefit from high resolution
- Large objects, high contrast, can do with lower resolution
- If both present at utions may be useful
- Local statistics suc ry in different parts of an image
- Exploit this in multiresolution proces

# Image Pyramids

- An image pyramid is a collection of decreasing resolution images arranged in the shape of a pyramid.

# Image Pyramids

System block diagram for creating image pyramids



1. Compute a reduced-resolution approxima image by filtering and downsampling (mean, Gaussian, subsamp
2. Upsample the output of step 1 and filter the result (possibly with interpolation)
3. Compute the difference between the prediction of step 2 and the input to step 1

Repeating, produce approximation and prediction residual pyramids

# Image Pyramids

Two image pyramids and their statistics (Gaussian approx pyramid, Laplacian prediction residual pyramid)



To recreate image

- Upsample and filter the lowest resolution approximation image
- Add the 1-level higher Laplacian's prediction residual

# References and Acknowledgement

- *Gonzalez* and Woods, 2002, *Chapter 4.1-4.4, 7.1*
- *Szeliski Chapter 3.1-3.5*
- Some material, including images and tables, were drawn from the text *g* by Gonzalez and Woods,