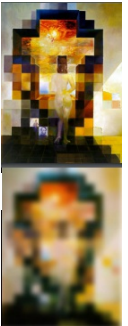


2C-L1 Fourier transform

2017/11/21 15:41

1. Sum
 - a. frequency aspect
 - b. via Fourier Transform to get into this aspect
 - c. DFT is the key, since images are discrete
 - d. the relationship between what's in the image and what's the frequency accordingly
2. Intro
 - a. previously
 - i. image as a function
 - ii. edges
 - iii. Hough transform to extract info. from images as evidence
 - b. Now
 - i. about image processing - frequency analysis
 - ii. 40 years ago, Electrical engineers define your filters and other types of **image analysis operation** in terms of what's called **frequency response**.
 1. how things change when you have high frequency stuff, and then low frequency stuff.
 - iii. we are more oriented to computer science, where we tend to **think of images as data structures** and not so much as a signal
 - iv. for a phenomena referred to as aliasing [混淆现象]. better to think about it from a **frequency perspective**

3. Dali

- a. look at the Dali's painting closely, you see lots of squares and details; when you look at it far away, it's Lincoln.
 - i. i.e. high frequency information that's showing you all these squares and other details. while at the low frequency, it actually is a reasonable picture of Abraham Lincoln
- b. The idea is any signal, e.g. images, can be decomposed into a set of things that describe it. And a basis set is what matters

1. Basis Sets

- a. definition
 - i. A basis **B** of a **vector space V** is a **linearly independent** subset of **V** that **spans V**.
 - ii. Theoretically, the basis only have to be linearly independent. But computationally, it's helpful to set basis orthogonal

2. Fourier

a. Meaning of basis set for images

i. useful for analysis especially for linear systems because we could consider each basis component independently.

1. because in linear systems things just sum. So if I could say how some operator applies to each element of the basis set, and then I just have to scale it and sum it, then I'd know how this operator applies to the entire image.

2. most of the operations are linear

b. How to define a suitable basis set

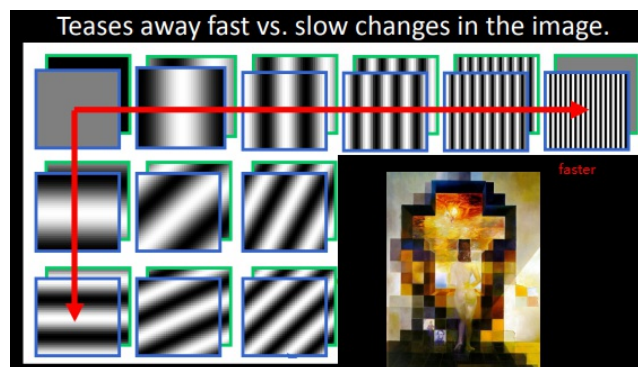
i. alternative 1:

- Consider an image as a point in a $N \times N$ size space – can rasterize into a single vector
$$[x_{00} x_{10} x_{20} \dots x_{(n-1)0} x_{10} \dots x_{(n-1)(n-1)}]^T$$
- The “normal” basis is just the vectors:
$$[0 \ 0 \ 0 \ 0 \dots 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \dots 0]^T$$

1. Valid, but not helpful since each one of these basis vectors is just a single pixel and the meaning behind it is how it behaves in respect to a single pixel

i. Alternative 2: a nice one

1. measure how quickly it varies in different directions



2. called Fourier basis, or Fourier decomposition

3. [Q]: how's related to Fourier Series

1. Fourier Series

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

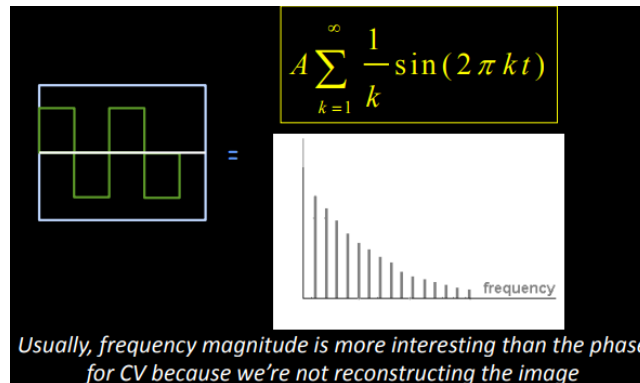
i. We're going to get from Fourier series to Fourier transform to discrete Fourier transform

b. A Sum of Sines

i. Our building block: $A \sin(\omega x + \phi)$

1. A: amplitude, a scale #the most important component
2. ω : frequency
3. ϕ : phase

c. Time and Frequency



i. time domain vs. frequency domain

ii. In Computer Vision, we're interested in how much power there is in each of the different frequencies. If we're actually going to reconstruct images then we would probably have to worry more about the phase to get you better reconstruction

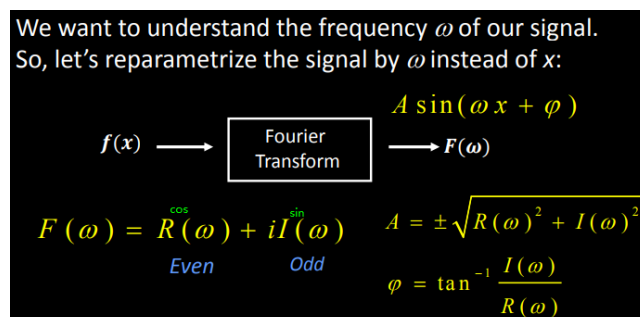
1. Fourier Transform

a. the motivation

i. to know how much power of any given frequency there is in an image.

1. so you need to transform our image from a function of time to in terms of its frequency

b. Fourier transform.



i. For every ω from 0 to ∞ (actually $-\infty$ to ∞), (A, ϕ) holds the amplitude A and phase ϕ of the corresponding sinusoid

1. by a complex number

1. Computing Fourier Transform

a. When we do a Fourier transform, all we're doing is computing a basis set.

b.

2. Fourier Transform More Formally

Fourier Transform – more formally

Represent the signal as an infinite weighted sum of an infinite number of sinusoids:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

Again: $e^{ik} = \cos k + i \sin k$ $i = \sqrt{-1}$

Spatial Domain (x) \longrightarrow Frequency Domain (ω or u or even s)
(Frequency Spectrum $F(u)$ or $F(\omega)$)

Inverse Fourier Transform (IFT) – add up all the sinusoids at x:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i2\pi ux} du$$

10. Frequency Spectra

a. the signal in the frequency axis with certain amplitude

Frequency actually goes from $-\infty$ to $+\infty$
Sinusoid example:

Even (cos) *Odd (sin)* *Magnitude*

Real *Imaginary* *Power*

11. Limitations

- The integral $\int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$ exists if the function f is integrable:

$$\int_{-\infty}^{\infty} |f(x)| dx < \infty$$

- If there is a bound of width T outside of which f is zero then obviously could integrate from just $-T/2$ to $T/2$

12. Fourier Transform to Fourier Series

Fourier Transform \Leftrightarrow Fourier Series

- The **Discrete FT**:

$$F(k) = \frac{1}{N} \sum_{x=0}^{x=N-1} f(x) e^{-i \frac{2\pi kx}{N}}$$

... where x is discrete and goes from the start of the signal to the end ($N-1$)

... and k is the number “cycles per period of the signal” or “cycles per image.”

- Only makes sense $k = -N/2$ to $N/2$. Why? What’s the highest frequency you can unambiguously have in a discrete image?

a. the highest frequency is $-N/2$ or $N/2$, the same.

1. 2D

2D Fourier Transforms

- The two dimensional version: .

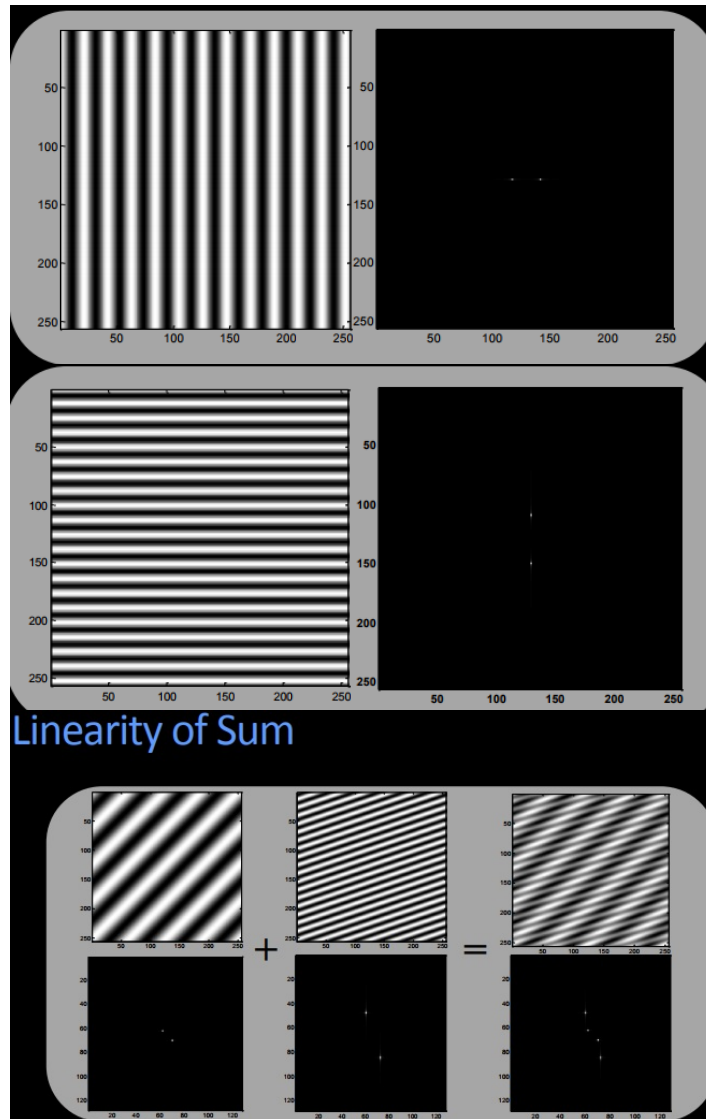
$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-i 2 \pi (ux + vy)} dx dy \frac{1}{2}$$

- And the 2D **Discrete FT**:

$$F(k_x, k_y) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-i \frac{2 \pi (k_x x + k_y y)}{N}}$$

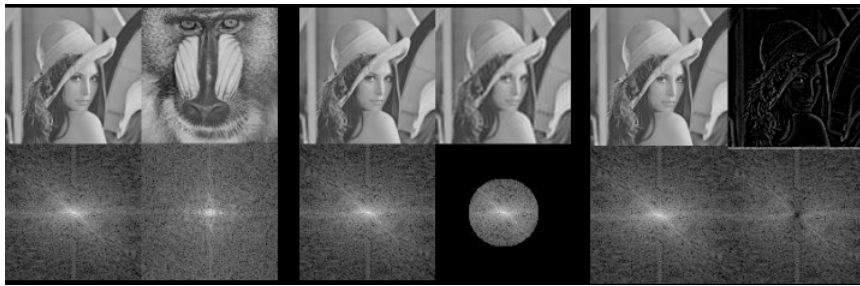
- Works best when you put the origin of k in the middle...

a. one example

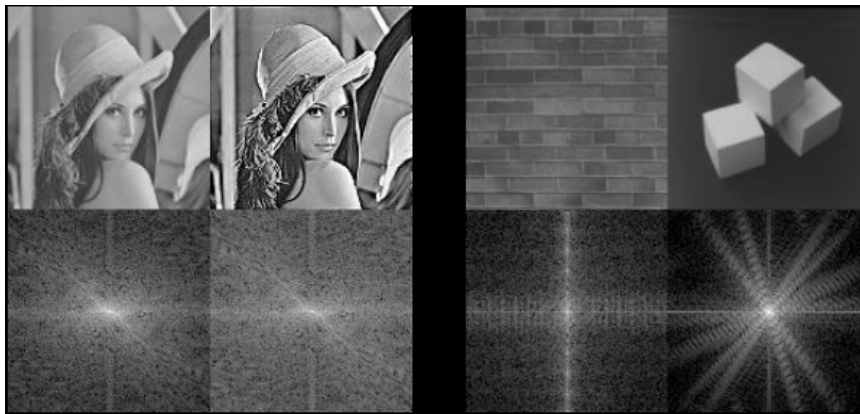


1. Fourier transfer, Fourier series is just made up of sums and multiplies, it's a linear operation. so the Fourier transform of the sum is just the sum of the Fourier transforms.

1. Examples in real images

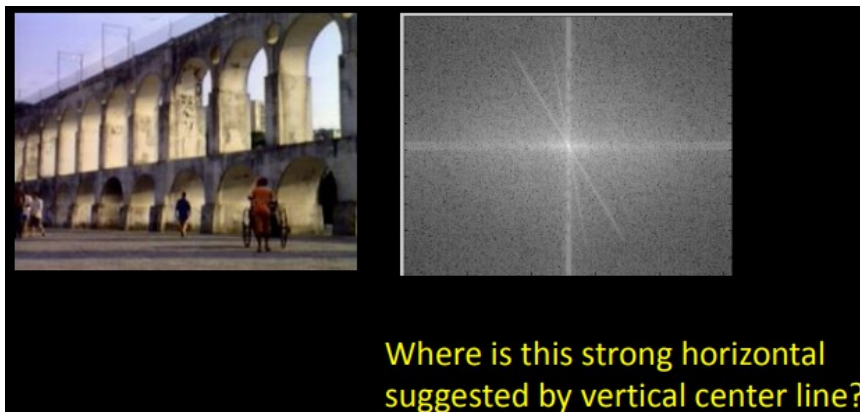


- a. the spectras of natural images are similar, the main difference is phase. But we're not reconstructing images so we only care about the frequency.
- b. to smooth the image is by remove the high frequency. The low frequency gives the main pattern image
- c. high frequency gives info. about edges. so to get the edges, we just have to remove the low frequency part.



- d. sharpen the image by $(2 * \text{img} - \text{smooth}(\text{img}))$. So sharp images have more high frequency components.
- e. Frequency shows the texture of the images. They have the same direction.

1. Man Made Scene



- a. in the spectra, there is a strong horizontal line and vertical line, which indicates one step(sharp) horizontal edge and one step vertical edge.
 - i. Basically, if you think about how the sinusoid goes in the image, it's essentially as if it assumes the image just kept wrapping around.
 - ii. So the top edge and the bottom edge are connected when applied with FT. And here comes this funny sort of sharp edge

- b. One way to eliminate that when you're doing **Fourier analysis** is you'll **multiply the whole image by a Gaussian** that tapers off towards zero by the edges, and then everything becomes nice and smooth.
 - i. and the Gaussian with wrap
 - c. When in doubt, fix your life with a Gaussian.
- 1. End
 - a. this lesson gives you some relation between sort of what's in the image, and the frequency components in it.
 - b. we're going to talk about how filtering and convolution are expressed in the frequency domain and that's how it allows us to get to aliasing.