Sampling

CS 510
Lecture #9
February 6, 2002

Goal: Image Manipulation

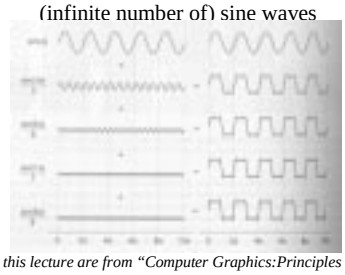
- Rotatation & Scale
- Filtering & reconstruction
- Compression
- Plane to plane projection
- Composition of two images
- Image to image matching

Today, we will introduce some sampling theory background

Please read Chapter 3 of Trucco & Verri for Friday

2D Sampling

Any repeating pattern can be constructed from an (infinite number of) sine waves



All figures in this lecture are from "Computer Graphics:Principles and Practice" by Foley, van Dam, Feiner & Hughes.

2D Fourier Spectrum

Any signal that is non-zero over a finite range can also be represented by an infinite number of sine waves:

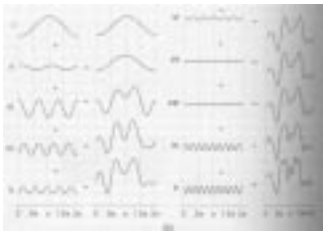


Figure shows reconstruction of one row of Mandrill image

Why?

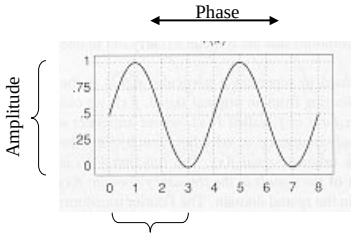
- Sine waves are a good representation for repeated patterns (e.g. visual textures such as bricks)
- Discrete pixel patterns are regular
  - Help us analyze what can/can't be in a given image
  - Help us manage artifacts
- Anytime compression
  - First few values approximate entire image
  - The more values, the more accuracy

Fourier Analysis ≠ Magic

- OK, many textbooks make is obscure, but…
- We are just rewriting a function f(x) over a finite range of x as a sum of sine waves
  - In effect, we pretend it repeats….
- For each sine wave, we specify:
  - A frequency
  - An amplitude
  - A phase

The Sine Wave

This may be a review from high school



Simplifying Phase

- Phase describes where the cycle crosses the x axis:
  - If it crosses at 0 and -π, it's a sine wave.
  - If it crosses at π/2 and - π/2, it a cosine wave.
  - In general, if it crosses at ϕ and ϕ + π radians, it has phase ϕ- π/2 (i.e., its based on cosine, not sine)
    - ϕ = 0 ⇒ cosine wave
    - ϕ = π ⇒ sine wave
- Phase seems to disappear …
- Any wave with phase ϕ can be expressed as: $\cos(x+\phi) = \alpha\cos(x) + \beta\sin(x)$

Phase (II)

Where:

$$\phi = \tan^{-1}\frac{\sum\beta}{\sum\alpha}$$

$$\cos(\theta+\phi) = \sqrt{\alpha^2\cos^2(\theta)+\beta^2\sin^2(\theta)}$$

$(\theta+\phi)$ still indicates that the cosine curve has been shifted by ϕ degrees

Fourier Transform

- Mathematically, the Fourier transform in 1D is:

$$F(u) = \int_{-\infty}^{+\infty} f(x)[\cos 2\pi ux - i\sin 2\pi ux]dx$$

  where u is a frequency, F(u) is the amplitude(s) at that frequency, and i is the square root of –1
- The magnitude at a frequency is:

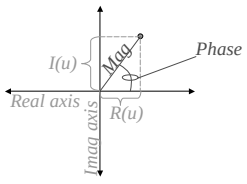
$$|F(u)| = \sqrt{R^2(u)+I^2(u)}$$

- The phase at a frequency is:

$$\tan^{-1}(u) = \frac{I(u)}{R(u)}$$

Maga[illegible]ag

- [illegible]
  a[illegible]ag[illegible]g[illegible] a[illegible]



Notation Warning

- You will also see the Fourier transform written as:

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

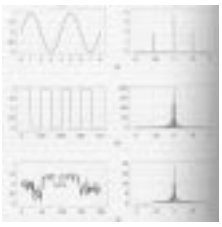
- This is equivalent, because of Euler's identity

$$e^{-i2\pi ux} = \cos(2\pi ux) - i\sin(2\pi ux)$$

- I will never use this form, however

Inverse Fourier

- The inverse mapping from frequencies (sines) to the spatial domain is:



$$f(x) = \int_{-\infty}^{+\infty} F(u)[\cos 2\pi ux + i \sin 2\pi ux]du$$

The DC component

- What happens when u = 0?
  - cosine(0) = 1
  - sine(0) = 0
- So

$$F(u) = \int_{-\infty}^{+\infty} (x)[\cos 2\pi ux - i \sin 2\pi ux]dx = \int_{-\infty}^{+\infty} (x)$$

This is the average value (or "DC component") of the function. For images, it is largely a function of lighting.

Discrete Fourier Transform

- Problem: an image is not an analogue signal that we can integrate. Therefore for 0 ≤u<N:

$$F(u) = \sum_{x=0}^{N-1} f(x) \cos \quad - i \sin$$

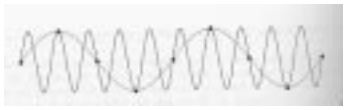
- And the discrete inverse transform is:

$$f(x) = \frac{1}{N} \sum_{x=0}^{N-1} F(u) \cos \quad + i \sin$$

Where N is the data size, f(x) are the data values

The Nyquist Rate

- What happened to the frequencies above N?
  - Above N, you have less than one sample per half-cycle
  - Therefore, high frequencies look like lower frequencies



- This is bad, very bad….

2D Fourier Transform

- So far, we have looked only at 1D signals
- For 2D signals, the Fourier transform is essentially the same:

$$F(u,v) \equiv \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)[\cos(2\pi(ux+vy)) - i \sin(2\pi(ux+vy))]$$

- Note that frequencies are now two-dimensional (u= freq in x, v = freq in y)
- For every frequency (u,v), there is a real and an imaginary value

Frequency Aliasing

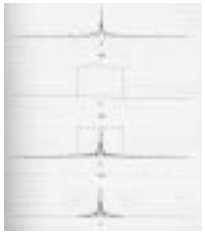
- The fact that high-frequency information masquerades as low-frequency information is called frequency aliasing.
- Low-pass filtering is important because it allows you to remove higher frequencies before reducing an image.
  - Example: reduce an image from 1000x1000 to 800x800
  - Nyquist rate of destination is lower than source

Low-Pass Filtering

- By definition, a low-pass filter zeroes out all the high frequencies in the Fourier spectrum

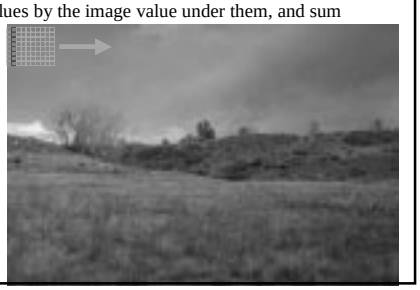
To low-pass filter an image:
1) convert to frequency domain
2) discard are values for u > thresh
3) convert image back to spatial domain

But is there an easier way?



Convolution

"Slide" mask over image. At each window position, multiply the mask values by the image value under them, and sum the results.



Convolution (II)

- Formally, convolution is often expressed as:

$$f(x)\otimes g(x)\equiv\int_{-\infty}^{+\infty}f(\tau)g(x-\tau)d\tau$$

- Of course, we are dealing with finite, discrete functions:

$$f(x)\otimes g(x)\equiv\sum_{i=-M/2}^{M/2}f(i)g(x-i)$$

Convolution Examples

- Let F1 = [1,2,3,4,5]
- Let F2 = [1,2,1,2,1]
- Let G1 = [-1,2,-1]
- Let G2 = [1/3,1/3,1/3]

- Then F1*G1=[0,0,0,0,6]
- Then F2*G1=[0,2,-2,2,0]
- Then F1*G2=[1,2,3,4,3]
- Then F2*G2=[1,4/3,5/3,4/3,1]

Convolution (III)

- Why introduce convolution now?

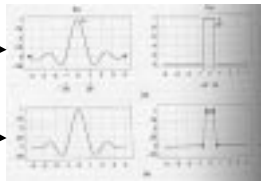
- Because multiplying two Fourier transforms in the frequency domain is the same as convolving their inverse Fourier transforms in the spatial domain! (trust me)

Low-Pass Filter (return)

- To Low-Pass filter, we can multiply by a pulse in frequency space, or
- Convolve the original image with the inverse Fourier transform of a pulse...

sinc filter →

Truncated sinc →

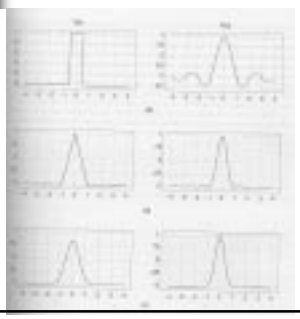


See T&V pg. 58, too

The Gibbs Phenomenon
(ringing)

- The truncated sinc is no longer a pulse in frequency space
  - passes small amounts of some high frequencies
  - passes acceptable frequencies in uneven amounts
  - may create negative values in unusual circumstances

Alternative Filters



pulse/sinc

triangle/$sinc^2$

gaussian/gaussian

Image Transformations

- Now we have the background to consider simple image transformations.
- There are always two components:
  - *Geometric*: finding which point in a source image is mapped to the center of a pixel in the target image
    algebraic, incremental methods
  - *Photometric*: computing the value of the target pixel
    filtering methods

Image Reductions

- Anytime the target image has a lower resolution than the source image, prevent frequency aliasing by low-pass filtering.
  - In practice, convolve with a Gaussian
  - Determine Nyquist rate for target image
  - Select σ
  - Convolve source image with $g(\sigma)$
  - Apply geometric transformation to result

Image Reductions (II)

- Example: reduce from 1Kx1K to 800x800 pixels
  - Select one (source) pixel as unit length
  - The Nyquist rate for source is 0.5 cycles/pixel
  - Nyquist rate for target is 0.4 cycles/(source)pixel
  - Problem: Gaussian is not a strict cut-off
    - Select "pass" value (2σ sounds good)
    - Select mask width to cover "most" of the area under the Gaussian curve
      - T&V recommend 5σ,
      - Covers 98.75% of the area under the Gaussian curve

Image Reduction (III)

- So 2σ is 0.4 cycles/pixel
  - The Fourier transform of $g(x, \sigma)$ is $g(\omega, 1/\sigma)$
  - The inverse of 0.4 cycles/pixel is 2.5 pixels/cycle
    - σ = 1.25 pixels/cycle
  - (T&V): To include 5σ of the curve, $\sigma = w/5$,
    - w is the width of the mask
    - W = 6.25
- So create a 7x7 Gaussian mask with sigma 1.25
  - w should be odd, so don't use 6x6
    - Why make w odd? To avoid a geometric transformation…
- Smooth the image using this mask, then subsample.

Image Transformation

- What if we want to keep the image 1Kx1K?
  - Target Nyquist rate is 0.5 cycles/pixel
  - In image space, 2σ = 2 pixels/cycle, so σ=1
  - σ = w/5, so w = 5
  - Create a 5x5 mask with σ=1, smooth source image
  - Transform (rotate, etc.) the result.
- This is why every image processing package includes predefined 5x5 Gaussian masks
  - E.g. Intel's IPL
- Other masks you build yourself

Smoothing with σ=1

Original Image

Image with Gaussian Smoothing, σ = 1.0

Limits to Gaussians

- The Gaussian mask itself is a discrete sampling of a continuous signal.
- Gaussian signals with sigmas below 0.8 are too small to be sampled at pixel intervals.
- Generally not used for "up-sampling"

Implications of Smoothing

- All of this is based on the idea of representing the image as a sum of sine waves.
- Physically, this assumption is absurd
  - Think of your ray tracer (or radiosity) -- where would sine waves (or repeating signals) come from?
  - Object edges lead to non-differentiable intensity jumps
    - the signal content on the two sides are unrelated
  - Edges are therefore very high frequency;
    - G(x, σ=1) blurs the image
- Fourier analysis does not describe image content -- but it does describe the limitations of A/D conversion, and therefore of image manipulation