The blue and green colors are actually the same
Hybrid Images

Why do we get different, distance-dependent interpretations of hybrid images?
Thinking in Frequency

Slides: Hoiem, Efros, and others
Recap of Filtering

• Linear filtering is dot product at each position
  – Not a matrix multiplication
  – Can smooth, sharpen, translate
    (among many other uses)

• Be aware of details for filter size, extrapolation, cropping
Median filters

• A **Median Filter** operates over a window by selecting the median intensity in the window.
• What advantage does a median filter have over a mean filter?
• Is a median filter a kind of convolution?
Comparison: salt and pepper noise

3x3

5x5

7x7
Review: questions

1. Write down a 3x3 filter that returns a positive value if the average value of the 4-adjacent neighbors is less than the center and a negative value otherwise.

2. Write down a filter that will compute the gradient in the x-direction:

   \[ \text{grad}_x(y,x) = \text{im}(y,x+1) - \text{im}(y,x) \] for each \( x, y \)
Review: questions

3. Fill in the blanks:

a) \_ = D \times B
b) A = \_ \times \_
c) F = D \times \_
d) \_ = D \times D
This lecture

• Fourier transform and frequency domain
  – Frequency view of filtering
• Reminder: Read your textbook
  – Today’s lecture covers material in 3.4
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?
Why does a lower resolution image still make sense to us? What do we lose?
Thinking in terms of frequency
<table>
<thead>
<tr>
<th>EXAMS</th>
<th>STUDY MATERIALS</th>
<th>RELATED RESOURCES</th>
<th>DOWNLOAD COURSE MATERIALS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lecture 9: Multiplication and inverse matrices</td>
<td>Lecture 4: Factorization into A = LU</td>
<td>Lecture 5: Transposes, permutations, spaces R^n</td>
</tr>
<tr>
<td></td>
<td>Lecture 6: Column space and nullspace</td>
<td>Lecture 7: Solving Ax = 0: pivot variables, special solutions</td>
<td>Lecture 8: Solving Ax = b: row reduced form R</td>
</tr>
<tr>
<td></td>
<td>Lecture 9: Independence, basis, and dimension</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Background: Change of Basis

For vectors and for image patches
How is it that a 4MP image can be compressed to a few hundred KB without a noticeable change?

Related concept: Image Compression
Lossy Image Compression (JPEG)

Block-based Discrete Cosine Transform (DCT)

https://en.wikipedia.org/wiki/JPEG
Using DCT in JPEG

- The first coefficient $B(0,0)$ is the DC component, the average intensity
- The top-left coeffs represent low frequencies, the bottom right – high frequencies
Lossy Image Compression (JPEG)

8x8 image patch

DCT bases

Patch representation after projecting on to DCT bases

\[ G = \begin{bmatrix}
-415.38 & -30.19 & -61.20 & 27.24 & 56.13 & -20.10 & -2.39 & 0.46 \\
-46.83 & 7.37 & 77.13 & -24.56 & -29.91 & 9.93 & 5.42 & -5.65 \\
12.1 & -6.55 & -13.20 & -3.95 & -1.88 & 1.75 & -2.79 & 3.14 \\
-7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\
-1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\
-0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \\
\end{bmatrix} \]
Image compression using DCT

• Quantize
  – More coarsely for high frequencies (which also tend to have smaller values)
  – Many quantized high frequency values will be zero

• Encode
  – Can decode with inverse dct

Filter responses

\[
G = \begin{bmatrix}
-415.38 & -30.19 & -61.20 & 27.24 & 56.13 & -20.10 & -2.39 & 0.46 \\
-46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\
-48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\
12.12 & -6.55 & -13.20 & -3.95 & -1.88 & 1.75 & -2.79 & 3.14 \\
-7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\
-1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\
-0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \\
\end{bmatrix}
\]

Quantization table

\[
Q = \begin{bmatrix}
16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\
12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\
14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\
14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\
18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\
24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\
49 & 94 & 78 & 87 & 103 & 121 & 120 & 101 \\
72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \\
\end{bmatrix}
\]
JPEG Compression Summary

1. Convert image to YCrCb
2. Subsample color by factor of 2
   - People have bad resolution for color
3. Split into blocks (8x8, typically), subtract 128
4. For each block
   a. Compute DCT coefficients
   b. Coarsely quantize
      • Many high frequency components will become zero
   c. Encode (e.g., with Huffman coding)

http://en.wikipedia.org/wiki/YCbCr
http://en.wikipedia.org/wiki/JPEG
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.
How would math have changed if the Slanket or Snuggie had been invented?
A sum of sines

Our building block:

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

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

$$f_{\text{target}} = f_1 + f_2 + f_3 + \ldots + f_n + \ldots$$
Frequency Spectra

- example: $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi (3f) t)$
Frequency Spectra
Frequency Spectra

= 

= 

= 

+ 

= 

= 

= 

=
Frequency Spectra
Frequency Spectra
Frequency Spectra
Frequency Spectra
Frequency Spectra

\[ A \sum_{k=1}^{\infty} \frac{1}{k} \sin(2\pi kt) \]

frequency
Example: Music

- We think of music in terms of frequencies at different magnitudes
Other signals

• We can also think of all kinds of other signals the same way

[Cartoon image of a conversation where one character says, "Hi, Dr. Elizabeth? Yeah, uh... I accidentally took the Fourier transform of my cat... Meow!"]
Fourier analysis in images

Intensity Image

Fourier Image

http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering
Fourier Transform

- Fourier transform stores the magnitude and phase at each frequency
  - Magnitude encodes how much signal there is at a particular frequency
  - Phase encodes spatial information (indirectly)
  - For mathematical convenience, this is often notated in terms of real and complex numbers

\[
A = \pm \sqrt{R(\omega)^2 + I(\omega)^2} \quad \text{Amplitude:} \quad \phi = \tan^{-1} \frac{I(\omega)}{R(\omega)} \quad \text{Phase:}
\]
Salvador Dali invented Hybrid Images?

Salvador Dali
“Gala Contemplating the Mediterranean Sea, which at 20 meters becomes the portrait of Abraham Lincoln”, 1976
Fourier Bases

Teases away fast vs. slow changes in the image.

This change of basis is the Fourier Transform.
Fourier Bases

Fourier domain with complex amplitude: $a+jb$

$\begin{align*}
a-jb \\
a+jb
\end{align*}$

Discrete Fourier Transform 13
This looks a lot like DCT in JPEG compression.
Man-made Scene
Can change spectrum, then reconstruct
Low and High Pass filtering
Computing the Fourier Transform

\[ H(\omega) = \mathcal{F} \{ h(x) \} = Ae^{j\phi} \]

Continuous

\[ H(\omega) = \int_{-\infty}^{\infty} h(x) e^{-j\omega x} \, dx \]

Discrete

\[ H(k) = \frac{1}{N} \sum_{x=0}^{N-1} h(x) e^{-j\frac{2\pi k x}{N}} \quad k = -N/2..N/2 \]

Fast Fourier Transform (FFT): NlogN
The Convolution Theorem

• The Fourier transform of the convolution of two functions is the product of their Fourier transforms

\[ F[g * h] = F[g] F[h] \]

• Convolution in spatial domain is equivalent to multiplication in frequency domain!

\[ g * h = F^{-1}[F[g] F[h]] \]
Filtering in spatial domain

\[
\begin{pmatrix}
1 & 0 & -1 \\
2 & 0 & -2 \\
1 & 0 & -1 \\
\end{pmatrix}
\]
Filtering in frequency domain

Slide: Hoiem
Filtering

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?
Gaussian
Box Filter
Is convolution invertible?

• If convolution is just multiplication in the Fourier domain, isn’t deconvolution just division?
• Sometimes, it clearly is invertible (e.g. a convolution with an identity filter)
• In one case, it clearly isn’t invertible (e.g. convolution with an all zero filter)
• What about for common filters like a Gaussian?
But you can’t invert multiplication by 0

- But it’s not quite zero, is it…
Let’s experiment on Novak
Convolution

\[ \text{Convolution} \]

\[ \text{FFT} \quad * \quad \text{FFT} = \text{iFFT} \]
Deconvolution?

\[
\text{iFFT} \uparrow = \text{FFT} \downarrow \div \text{FFT} \downarrow
\]
But under more realistic conditions

Random noise, .000001 magnitude
But under more realistic conditions

Random noise, .0001 magnitude
But under more realistic conditions

\[ \text{Random noise, .001 magnitude} \]
With a random filter...

Random noise, .001 magnitude
Deconvolution is hard

- Active research area.
- Even if you know the filter (non-blind deconvolution), it is still very hard and requires strong *regularization*.
- If you don’t know the filter (blind deconvolution) it is harder still.
Figure 1. Algorithm pipeline. Our algorithm iterates between $x$-step and $k$-step with the help of a patch prior for edge refinement process. In particular, we coerce edges to become sharp and increase local contrast for edge patches. The blur kernel is then updated using the strong gradients from the restored latent image. After kernel estimation, the method of [20] is used for final non-blind deconvolution.
Edge-based Blur Kernel Estimation Using Patch Priors.
Libin Sun, Sunghyun Cho, Jue Wang, and James Hays.
IEEE International Conference on Computational Photography 2013.