Does everyone have an override code?
Project 1 due Friday 9pm
Review of Filtering

• Filtering in frequency domain
  – Can be faster than filtering in spatial domain (for large filters)
  – Can help understand effect of filter
  – Algorithm:
    1. Convert image and filter to fft (fft2 in matlab)
    2. Pointwise-multiply ffts
    3. Convert result to spatial domain with ifft2

Did anyone play with the code?
Review of Filtering

• Linear filters for basic processing
  – Edge filter (high-pass)
  – Gaussian filter (low-pass)

\[
[-1, 1]
\]
More Useful Filters

Single Gaussian

1st Derivative of Gaussian

2nd Derivative (Laplacian of Gaussian)
Things to Remember

• Sometimes it makes sense to think of images and filtering in the frequency domain
  – Fourier analysis

• Can be faster to filter using FFT for large images
  • $N \log N$ vs. $N^2$ for auto-correlation

• Images are mostly smooth
  – Basis for compression

• Remember to low-pass before sampling
  • Otherwise you create aliasing
Aliasing and Moiré patterns

Gong 96, 1932, Claude Tousignant, Musée des Beaux-Arts de Montréal
The blue and green colors are actually the same

Why do we get different, distance-dependent interpretations of hybrid images?
Clues from Human Perception

- Early processing in humans filters for orientations and scales of frequency.
Perceptual cues in the mid-high frequencies dominate perception.
Application: Hybrid Images

When we see an image from far away, we are effectively subsampling it!

• A. Oliva, A. Torralba, P.G. Schyns, “Hybrid Images,” SIGGRAPH 2006
How is it that a 4MP image can be compressed to a few hundred KB without a noticeable change?
Lossy Image Compression (JPEG)

8x8 blocks

The first coefficient $B(0,0)$ is the DC component, the average intensity.

The top-left coeffs represent low frequencies, the bottom right represent high frequencies.

Block-based Discrete Cosine Transform (DCT)
Image compression using DCT

• Compute DCT filter responses in each 8x8 block

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

• Quantize to integer (div. by magic number; round)
  – More coarsely for high frequencies (which also tend to have smaller values)
  – Many quantized high frequency values will be zero

Quantization divisors (element-wise)

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

Quantized values

\[
B = \begin{bmatrix}
-26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\
0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\
-3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\
-3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]
JPEG Encoding

• Entropy coding (Huffman-variant)

Quantized values

\[
B = \begin{bmatrix}
-26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\
0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\
-3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\
-3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Linearize \( B \) like this.

Helps compression:
- We throw away the high frequencies (‘0’).
- The zig zag pattern increases in frequency space, so long runs of zeros.
Color spaces: YCbCr

Fast to compute, good for compression, used by TV

Y=0

Y=0.5

Y=1

Cb

Cr

James Hays
Most JPEG images & videos subsample chroma

- PSP Comp 3
  - 2x2 Chroma subsampling
  - 285K

- Original
  - 1,261K lossless
  - 968K PNG
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 (with run length encoding and then Huffman coding for leftovers)

http://en.wikipedia.org/wiki/YCbCr
http://en.wikipedia.org/wiki/JPEG
EDGE / BOUNDARY DETECTION
Szeliski 4.2
Edge detection

- **Goal:** Identify visual changes (discontinuities) in an image.

- Intuitively, semantic information is encoded in edges.

- What are some ‘causes’ of visual edges?

Source: D. Lowe
Origin of Edges

- Edges are caused by a variety of factors

Source: Steve Seitz
Why do we care about edges?

- Extract information
  - Recognize objects
- Help recover geometry and viewpoint
Closeup of edges
Closeup of edges
Closeup of edges

Source: D. Hoiem
Closeup of edges

Source: D. Hoiem
Characterizing edges

• An edge is a place of rapid change in the image intensity function.
Intensity profile
With a little Gaussian noise

Source: D. Hoiem
Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

Where is the edge?

Source: S. Seitz
Effects of noise

• Difference filters respond strongly to noise
  – Image noise results in pixels that look very different from their neighbors
  – Generally, the larger the noise the stronger the response

• What can we do about it?

Source: D. Forsyth
Solution: smooth first

- To find edges, look for peaks in \( \frac{d}{dx} (f * g) \)

Source: S. Seitz
Derivative theorem of convolution

- Differentiation is convolution, and convolution is associative:
  \[ \frac{d}{dx} (f * g) = f * \frac{d}{dx} g \]

- This saves us one operation:

Source: S. Seitz
Derivative of 2D Gaussian filter

* \[ [1 \ -1] = \]

Hays
Tradeoff between smoothing and localization

• Smoothed derivative removes noise, but blurs edge. Also finds edges at different “scales”.

Source: D. Forsyth
Think-Pair-Share

• What is a good edge detector?

• Do we lose information when we look at edges? Are edges ‘incomplete’ as a representation of images?
Designing an edge detector

• Criteria for a good edge detector:
  – **Good detection:** the optimal detector should find all real edges, ignoring noise or other artifacts
  – **Good localization**
    • the edges detected must be as close as possible to the true edges
    • the detector must return one point only for each true edge point

• Cues of edge detection
  – Differences in color, intensity, or texture across the boundary
  – Continuity and closure
  – High-level knowledge

Source: L. Fei-Fei
Designing an edge detector

• “All real edges”
  • We can aim to differentiate later on which edges are ‘useful’ for our applications.
  • If we can’t find all things which could be called an edge, we don’t have that choice.

• Is this possible?
Closeup of edges

Source: D. Hoiem
Figure 2. The problem of local estimation scale. Different structures in a natural image require different spatial scales for local estimation. The original image contains edges over a broad range of contrasts and blur scales. In the middle are shown the edges detected with a Canny/Deriche operator tuned to detect structure in the mannequin. On the right is shown the edges detected with a Canny/Deriche operator tuned to detect the smooth contour of the shadow. Parameters are $(\alpha = 1.25, \omega = 0.02)$ and $(\alpha = 0.5, \omega = 0.02)$, respectively. See (Deriche, 1987) for details of the Deriche detector.

What information would we need to ‘invert’ the edge detection process?
Elder – Are Edges Incomplete? 1999

Edge ‘code’:
- position,
- gradient magnitude,
- gradient direction,
- blur.

Figure 8. Top left: Original image. Top right: Detected edge locations. Middle left: Intermediate solution to the heat equation. Middle right: Reconstructed luminance function. Bottom left: Reblurred result. Bottom right: Error map (reblurred result—original). Bright indicates overestimation of intensity, dark indicates underestimation. Edge density is 1.7%. RMS error is 10.1 grey levels, with a 3.9 grey level DC component, and an estimated 1.6 grey levels due to noise removal.
Where do humans see boundaries?

- Berkeley segmentation database:
  
  [http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/)
pB boundary detector

Martin, Fowlkes, Malik 2004: Learning to Detect Natural Boundaries…
pB Boundary Detector

Figure from Fowlkes
Results

Human (0.95)

Pb (0.88)
Results

Human (0.96)

Pb (0.88)
For more:
45 years of boundary detection

Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011 (pdf)
State of edge detection

• Local edge detection works well
  – ‘False positives’ from illumination and texture edges (depends on our application).

• Some methods to take into account longer contours

• Modern methods that actually “learn” from data.

• Poor use of object and high-level information.
Wednesday

• Classic Canny edge detector – 22,000 citations
• Interest Points and Corners