Lecture 3:
The Camera Image Processing Pipeline (part 2)
Visual Computing Systems
CMU 15-769, Fall 2016
## Simplified image processing pipeline

- Correct for sensor bias (using measurements of optically black pixels)
- Correct pixel defects
- Vignetting compensation
- Dark-frame subtract (optional)
- White balance

- Demosaic
- Denoise / sharpen, etc.
- Color Space Conversion

- Gamma Correction (Non-linear mapping)
- Color Space Conversion (Y’CbCr)

- Chroma Subsampling
- JPEG compression

### Pixel values and bit depths:

- **12-bits per pixel**
  - 1 intensity per pixel
  - Pixel values linear in energy

- **3x12-bits per pixel**
  - RGB intensity per pixel
  - Pixel values linear in energy

- **3x8-bits per pixel**
  - Pixel values **perceptually** linear
JPG Compression
JPEG compression: the big ideas

- Low-frequency content is predominant in images of the real world

- The human visual system is:
  - less sensitive to high frequency sources of error
  - less sensitive to detail in chromaticity than in luminance

Therefore, it’s often acceptable for a compression scheme to introduce errors in high-frequency components of the image.

[Credit: Pat Hanrahan]
**Y’CbCr color space**

Y’ = luma: perceived luminance (non-linear)
Cb = blue-yellow deviation from gray
Cr = red-cyan deviation from gray

Conversion from R’G’B’ to Y’CbCr:

\[
Y' = 16 + \frac{65.738 \cdot R'_D}{256} + \frac{129.057 \cdot G'_D}{256} + \frac{25.064 \cdot B'_D}{256}
\]

\[
C_B = 128 + \frac{-37.945 \cdot R'_D}{256} - \frac{74.494 \cdot G'_D}{256} + \frac{112.439 \cdot B'_D}{256}
\]

\[
C_R = 128 + \frac{112.439 \cdot R'_D}{256} - \frac{94.154 \cdot G'_D}{256} + \frac{18.285 \cdot B'_D}{256}
\]
Example: compression in Y’CbCr

Original picture of Kayvon
Example: compression in Y’CbCr

Contents of CbCr color channels downsampled by a factor of 20 in each dimension
(400x reduction in number of samples)
Example: compression in Y’CbCr

Full resolution sampling of luma (Y’)

CMU 15-769, Fall 2016
Example: compression in Y’CbCr

Reconstructed result
(looks pretty good)
# Chroma subsampling

Y’CbCr is an efficient representation for storage (and transmission) because Y’ can be stored at higher resolution than CbCr without significant loss in perceived visual quality.

<table>
<thead>
<tr>
<th>Y'00</th>
<th>Y'10</th>
<th>Y'20</th>
<th>Y'30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cb00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cr00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Y'01</th>
<th>Y'11</th>
<th>Y'21</th>
<th>Y'31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cb01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cr01</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 4:2:2 representation:
- Store Y’ at full resolution
- Store Cb, Cr at full vertical resolution, but only half horizontal resolution

### 4:2:0 representation:
- Store Y’ at full resolution
- Store Cb, Cr at half resolution in both dimensions

### X:Y:Z notation:
- X = width of block
- Y = number of chroma samples in first row
- Z = number of chroma samples in second row
Apply discrete cosine transform (DCT) to each 8x8 block of image values

\[
basis[i, j] = \cos\left(\frac{\pi i}{N}(x + \frac{1}{2})\right) \times \cos\left(\frac{\pi j}{N}(y + \frac{1}{2})\right)
\]

DCT computes projection of image onto 64 basis functions: basis[i, j]

DCT applied to 8x8 pixel blocks of Y' channel, 16x16 pixel blocks of Cb, Cr (assuming 4:2:0)
Quantization

Quantization produces small values for coefficients (only few bits needed per coefficient)
Notice: quantization zeros out many coefficients

Result of DCT
(representation of image in cosine basis)

\[
\begin{bmatrix}
-415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\
4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\
-47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\
-49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\
12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\
-8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\
-1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\
0 & 0 & -1 & -4 & -1 & 0 & 1 & 2
\end{bmatrix}
\]

Quantization Matrix

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

Result of DCT

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

Changing JPEG quality setting in your favorite photo app modifies this matrix (“lower quality” = higher values for elements in quantization matrix)

Quantization produces small values for coefficients (only few bits needed per coefficient)
Notice: quantization zeros out many coefficients

Slide credit: Wikipedia, Pat Hanrahan
CMU 15-769, Fall 2016
JPEG compression artifacts

- Noticeable 8x8 pixel block boundaries
- Noticeable error near large color gradients
- Low-frequency regions of image represented accurately even under high compression
JPEG compression artifacts

Why might JPEG compression not be a good compression scheme for illustrations and rasterized text?
Lossless compression of quantized DCT values

Quantized DCT Values

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

Entropy encoding: (lossless)

Reorder values

Run-length encode (RLE) 0’s

Huffman encode non-zero values

JPEG compression summary

Convert image to Y’CbCr

Downsample CbCr (to 4:2:2 or 4:2:0)  (information loss occurs here)

For each color channel (Y’, Cb, Cr):

  For each 8x8 block of values
    Compute DCT
    Quantize results  (information loss occurs here)
    Reorder values
    Run-length encode 0-spans
    Huffman encode non-zero values
Key theme: exploit characteristics of human perception to build efficient image storage and image processing systems

- Separation of luminance from chrominance in color representation ($Y'CrCb$) allows reduced resolution in chrominance channels (4:2:0)

- Encode pixel values linearly in lightness (perceived brightness), not in luminance (distribute representable values uniformly in perceptual space)

- JPEG compression significantly reduces file size at cost of quantization error in high spatial frequencies
  - Human brain is more tolerant of errors in high frequency image components than in low frequency ones
  - Images of the real-world are dominated by low-frequency components
Auto Focus / Auto Exposure
Autofocus demos

- Phase-detection auto focus
  - Common in SLRs

- Contrast-detection auto focus
  - Point-and-shoots, smart-phone cameras
SLR Camera

- Pentaprism
- Autoexposure (AE)
- Viewfinder
- Focusing screen
- Autofocus (AF)

Image credits: Nikon, Marc Levoy
Nikon D7000

- Auto-focus sensor: 39 regions
- Metering sensor: 2K pixels
  - Auto-exposure
  - Auto-white-balance
  - Subject tracking to aid focus (predicts movement)
- Shutter lag ~ 50ms
Auto exposure

Low resolution metering sensor capture

Metering sensor pixels are large
(higher dynamic range than main sensor)

How do we set exposure?

What if a camera doesn’t have a separate metering sensor?

Image credits: Marc Levoy, Andrew Adams
AF/AE summary

- DSLRs have additional sensing/processing hardware to assist with the "3A's" (auto-focus, auto-exposure, auto-white-balance)
  - Phase-detection AF: optical system directs light to AF sensor
  - Example: Nikon metering sensor: large pixels to avoid over-saturation

- Point-and-shoots/smartphone cameras make these measurements by performing image processing operations on data from the main sensor
  - Contrast-detection AF: search for lens position that produces large image gradients
  - Exposure metering: if pixels are saturating, meter again with lower exposure

- In general, implementing AF/AE/AWB is an image understanding problem ("computer vision")
  - Understand the scene well enough to set the camera’s image capture and image processing parameters to best approximate the image a human would perceive
  - As processing/sensing capability increases, algorithms are becoming more sophisticated
High-dynamic range images

- Problem: ratio of brightest object to darkest object in real-world scenes can be quite large
  - Human eye can discern ratio of 100,000:1 (even more if accounting for adaptation)

- High-dynamic range (HDR) image: encodes large range of luminance (or lightness) values
  - Common format: 16-bits per channel EXR (see environment maps in Asst. 3)

- Modern camera senses can only sense much narrower range of luminances (e.g., 12-bit pixels)

- But most modern displays can only display a much narrower range of luminances
  - Luminance of white pixel / luminance of black pixel for a high-end LCD TV ~ 3000:1 *

* Ignore most marketing specs, which are now claiming over 2,000,000:1

Overexposed (loss of detail in brightest areas since they are clamped to 1)

Underexposed (detail remains in brightest areas, but large regions of image clamped to 0)
**Tone mapping**

- Tone mapping: non-linear mapping of wide range of luminances into a narrower range (for storage in low-bit depth image, or for presentation on a low-dynamic range display)
  - For examples see Debevec 1997, Reinhard 2002, Fattal 2002

- How to acquire HDR images with conventional camera?
  - Take multiple photos (often at multiple exposures), combine into a single HDR image

![Low-dynamic range images taken at multiple exposures](image1)

![Low-dynamic range image that is result of tone mapping HDR image](image2)
HDR “mode” in modern cameras

Kayvon’s iPhone
Smarter cameras

Goal: help photographer capture the shot they want

Face detection: camera finds faces: tunes AWB, AE, AF for these regions

Another example: iPhone burst mode “best” shot selection

Sony’s ill-fated “smile shutter”
Camera detects smile and automatically takes picture.
Smarter cameras

- Future behaviors
  - Automatic photo framing/cropping?
  - Replace undesirable data with more desirable data acquired previously

"Face-swapping"  
[Bitouk et al. 2008]

Result: Composite image with everyone’s eyes open

Four source photos:
in each shot, at least one child’s eyes are closed
Takeways

- The values of pixels in photograph you see on screen are quite different than the values output by the photosensor in a modern digital camera.

- The sequence of operations we discussed today is carried out at high frame rates by the image signal processing ASIC in most cameras today.

- In the coming lectures we’ll discuss more advanced image processing operations that are emerging in modern camera pipelines.
  - Local contrast enhancement, advanced denoising, high-dynamic range imaging, etc.
  - Growing sophistication and diversity of techniques suggests that current ISPs will likely become more programmable in the near future.

Qualcomm Snapdragon 820
Image Signal Processor (ISP): ASIC for processing pixels off camera (25MP at 30Hz)