Introduction to Image Processing

Professor Deepa Kundur
University of Toronto

Analog Intensity Images

- **continuous-space** and **continuous-amplitude** image consisting of intensity (grayscale) values
- \( I(x, y) \) is a two-dimensional signal representing the grayscale value at location \((x, y)\) where:
  - \( 0 \leq x \leq L_x \) and \( 0 \leq y \leq L_y \)
  - \( I(x, y) = 0 \) represents black
  - \( I(x, y) = 1 \) represents white
  - \( 0 < I(x, y) < 1 \) represents proportional gray-value

The image shown is "Dixie Queens" (two schoolgirls at lunch from Hadleyville, Oregon, circa 1911), Roy C. Andrews collection, PH003-P954, Special Collections and University Archives, University of Oregon, Eugene, Oregon 97403-1299.
Analog Intensity Images

- $I(x, y)$ can be displayed as an intensity image or as a mesh graph.

Discrete-Space Intensity Images

- discrete-space and continuous-amplitude image consisting of intensity (grayscale) values
- $I(m, n)$ is a two-dimensional signal representing the grayscale value at location $(m, n)$ where:
  - $m = 0, 1, \ldots, N_x - 1$ and $n = 0, 1, \ldots, N_y - 1$
- $I(m, n) = 0$ represents black
- $I(m, n) = 1$ represents white
- $0 < I(m, n) < 1$ represents proportional gray-value

Digital Images

- discrete-space and discrete-amplitude
- $m = 0, 1, \ldots, N_x - 1$ and $n = 0, 1, \ldots, N_y - 1$
- image consisting of grayscale colors from a finite set $C$ and indexed via the set: $\{0, 1, 2, \ldots, N_C - 1\}$
- Example: $N_C = 8$ and grayscale values linearly distributed in intensity between black (0) and white ($N_C - 1$)
Digital Images: Common Format

- $I(m, n)$ is a two-dimensional signal representing the grayscale value at location $(m, n)$ where:
  - $I(m, n) \in \{0, 1, 2, \ldots, N_C - 1\}; \ N_C = \text{no. of colors}$
  - $I(m, n) = 0$ represents black
  - $I(m, n) = N_C - 1$ represents white
  - $I(m, n) \in \{1, 2, \ldots, N_C - 2\}$ represents proportional gray-value

Digital Images: 8-Bit Grayscale Images

- Standard 8-bit images use color indices from 0 through 255 to cover shades of gray ranging from black to white (inclusive).
  - convenient for programming: color representation occupies a single byte
  - perceptually acceptable: barely sufficient precision to avoid visible banding

Digital Images: Common Format

- $N_C$ is usually of the form $2^N$, so that the $2^N$ different colors are efficiently represented with $N$-bit binary notation; Example: $N = 3$

Digital Images: Color
Digital Images: Color

ADDITIVE COLOR

SUBTRACTIVE COLOR

Introduction to Image Processing

Color Spaces

▶ **Color space:** model describing a way to represent colors as mathematical vectors
▶ usually three or four numbers are needed to represent any color; common color spaces include:
  ▶ red (R), green (G), blue (B) popular for LCD displays
  ▶ cyan (C), magenta (M), yellow (Y), key (K) popular for print
  ▶ YCbCr, HSV, . . .

Digital Images: Additive Color Theory
Digital Images: Truecolor Images

- From Wiki (March 18, 2013): method of representing and storing graphical image information (especially in computer processing) in an RGB color space such that a very large number of colors, shades, and hues can be displayed in an image, such as in high quality photographic images or complex graphics.

- Usually at least 256 shades of each red, green and blue are employed, resulting in at least $256^3 = 16,777,216$ (16 million) color variations.

- Human eye can discern as many as ten million colors, so representation should exceed human visual system (HVS) capabilities!

RGB versus Grayscale

- RGB to grayscale conversion:

$$I(m, n) = 0.299R(m, n) + 0.587G(m, n) + 0.114B(m, n)$$

- Note: $0.299 + 0.587 + 0.114 = 1$.
- The luminance compensates for the eye’s distinct sensitivity to different colors.
- The human eye is most sensitive to green, then red, and last blue.
  - There are evolutionary justifications for this difference.
  - A color with more green is brighter to the eye than a color with more blue.
**RGB versus Grayscale**

- RGB to grayscale conversion:

\[ I(m, n) = 0.299R(m, n) + 0.587G(m, n) + 0.114B(m, n) \]

**Image Parameters**

- The following parameters have an effect on the image quality:
  - **sampling rate**: spatial resolution or dimension of image
  - **color depth**: number of colors or number of bits to represent colors

**Sampling Rate and Subsampling**
Color Depth and Amplitude Quantization

Lowpass Filtering

\[ H = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \]

\[ I_{\text{H}}(m, n) = I(m, n) \ast H(m, n) \]

Lowpass Filtering

\[ I(m, n) \text{ and } I_{H}(m, n): \]

Highpass Filtering

\[ H = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \]

\[ I_{\text{H}}(m, n) = I(m, n) \ast H(m, n) \]
Highpass Filtering

$I(m, n)$ and $I_H(m, n)$:

![Original Image](image1)

![Highpass Filtered Image](image2)

Edge Enhancement

$H = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$

$I_H(m, n) = I(m, n) * H(m, n)$

$I_E(m, n) = I_H(m, n) + I(m, n)$

2-D Discrete Fourier Transform

$\mathcal{F}(U, V) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} I(m, n)e^{-j2\pi(Um+Vn)}$

$I(m, n)$:
2-D Discrete Fourier Transform

\[ I_F(U, V): \]

2-D Discrete Cosine Transform

Consider an \( N_x \times N_y \)-dimensional digital image \( I(m, n) \):

\[ I_{DCT}(k, l) = \sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} I(m, n) \cos \left( \frac{\pi}{N_x} \left( n + \frac{1}{2} \right) k \right) \cos \left( \frac{\pi}{N_y} \left( m + \frac{1}{2} \right) l \right) \]
2-D Discrete 8 $\times$ 8 Cosine Transform

For $k, l \in \{0, 1, 2, \ldots, 7\}$,

$$l^B(m, n) = \sum_{k=0}^{7} \sum_{l=0}^{7} \alpha(k) \alpha(l) I^B_{DCT}(k, l) \cos \left[ \frac{\pi}{8} \left( n + \frac{1}{2} \right) k \right] \cos \left[ \frac{\pi}{8} \left( m + \frac{1}{2} \right) l \right]$$

where

$$\alpha(k) = \begin{cases} \sqrt{\frac{2}{8}} & \text{for } k = 0 \\ \sqrt{\frac{2}{8}} & \text{for } k = 1, 2, \ldots, 7 \end{cases}$$
Lossy versus Non-lossy Compression for Digital Images

- **Lossy compression**: remove signal components to reduce storage requirements
  - often exploits perceptual irrelevancy to shape the signal in order to reduce storage size
  - process is not reversible

- **Non-lossy compression**: exploit statistical redundancy to employ efficient codes (on average) to reduce storage requirements
  - process is reversible

---

**Lossy Compression via the DCT**

**Step 1**: Compute the $8 \times 8$-block DCT on $I(m, n)$.

\[
I_{DCT}^B(k, l) = \sum_{m=0}^{7} \sum_{n=0}^{7} I^B(m, n) \cos \left[ \frac{\pi}{8} \left( n + \frac{1}{2} \right) k \right] \cos \left[ \frac{\pi}{8} \left( m + \frac{1}{2} \right) l \right]
\]
Lossy Compression via the DCT

Step 1: Compute the $8 \times 8$-block DCT on $I(m, n)$.

$I^B(m, n)$ and $I^B_{DCT}(k, l)$:

Note: images displayed on log-amplitude scale.

Step 2: Remove high-frequency components via $R \times R$ mask.

$I^B_{DCT}(k, l)$ and compressed version $\tilde{I}^B_{DCT}(k, l)$ for $R = 4$:

Step 3: Compute the $8 \times 8$-block IDCT on compressed DCT coefficients.

$\tilde{I}^B(m, n) = \sum_{k=0}^{7} \sum_{l=0}^{7} \alpha(k) \alpha(l) \tilde{I}^B_{DCT}(k, l) \cos \left[ \frac{\pi}{8} \left( n + \frac{1}{2} \right) k \right] \cos \left[ \frac{\pi}{8} \left( m + \frac{1}{2} \right) l \right]$
Lossy Compression via the DCT

Step 3: Compute the $8 \times 8$-block IDCT on compressed DCT coefficients.

$\tilde{I}_{DCT}^R(k, l)$ and $\tilde{I}(m, n)$ for $R = 4$:

Further Compression Gains

- coefficients within the mask can be quantized with a factor determined by tests on human perception

$R \times R$ Square Mask

- compressed coefficients are passed through a non-lossy arithmetic coder for additional compression efficiency

Lossy Compression Results