Introduction to Image Processing

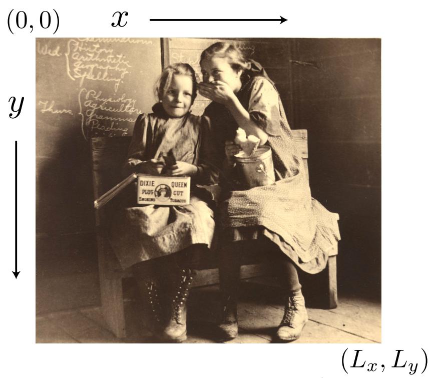
Dr. Deepa Kundur

University of Toronto

Analog Intensity Images

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.

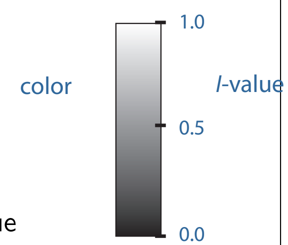
Analog Intensity Images



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.

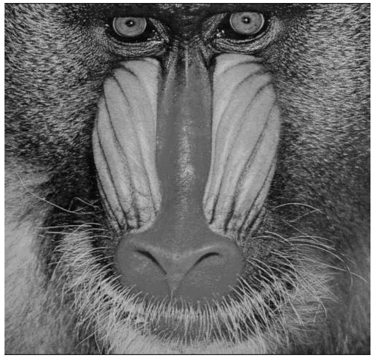
Analog Intensity Images

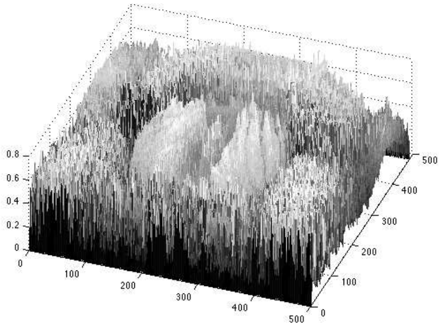
- continuous-space and continuous-amplitude image consisting of intensity (grayscale) values
- $I(x, y)$ is a two-dimensional signal representing the grayscale value at location (x, y) where:
- $0 \leq x \leq L_x$ and $0 \leq y \leq L_y$
- $I(x, y) = 0$ represents black
- $I(x, y) = 1$ represents white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



Analog Intensity Images

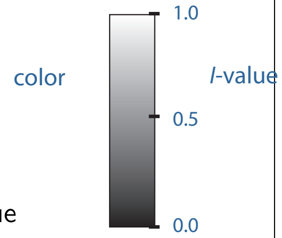
- $I(x, y)$ can be displayed as an intensity image or as a mesh graph





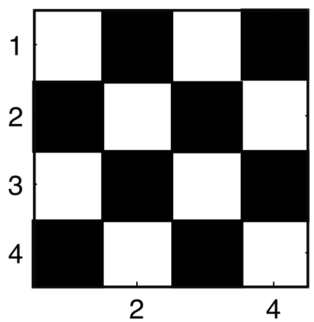
Discrete-Space Intensity Images

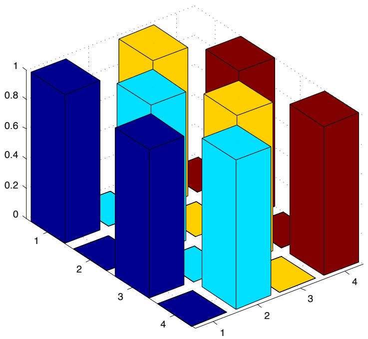
- discrete-space and continuous-amplitude image consisting of intensity (grayscale) values
- $I(m, n)$ is a two-dimensional signal representing the grayscale value at location (m, n) where:
- $m = 0, 1, \ldots, N_x - 1$ and $n = 0, 1, \ldots, N_y - 1$
- $I(m, n) = 0$ represents black
- $I(m, n) = 1$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



Discrete-Space Intensity Images

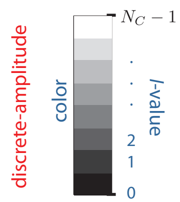
Example: 4×4 Checkerboard



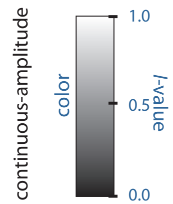


Digital Images

- 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 $\mathcal{C}$ 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 $(N_C - 1)$



. . . vs.

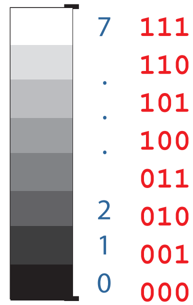


Digital Images: Common Format

- $I(m,n)$ 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 = 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: 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: $N=3$

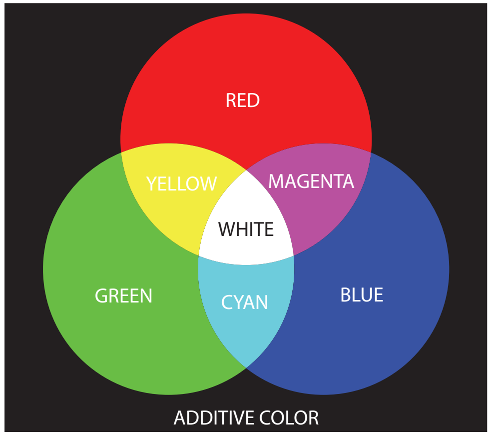


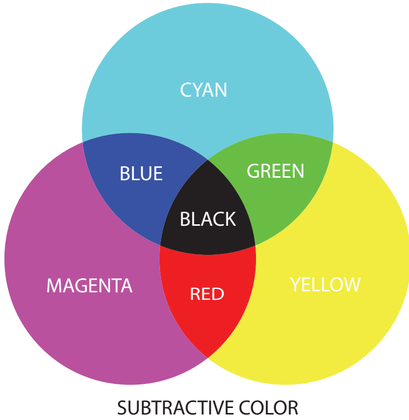
Digital Images: 8-Bit Grayscale Images

- 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: Color

Digital Images: Color

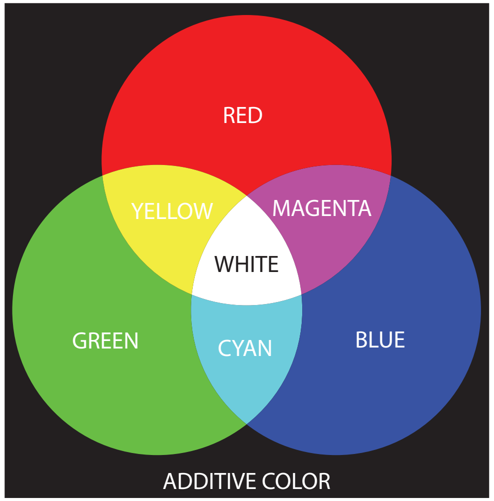


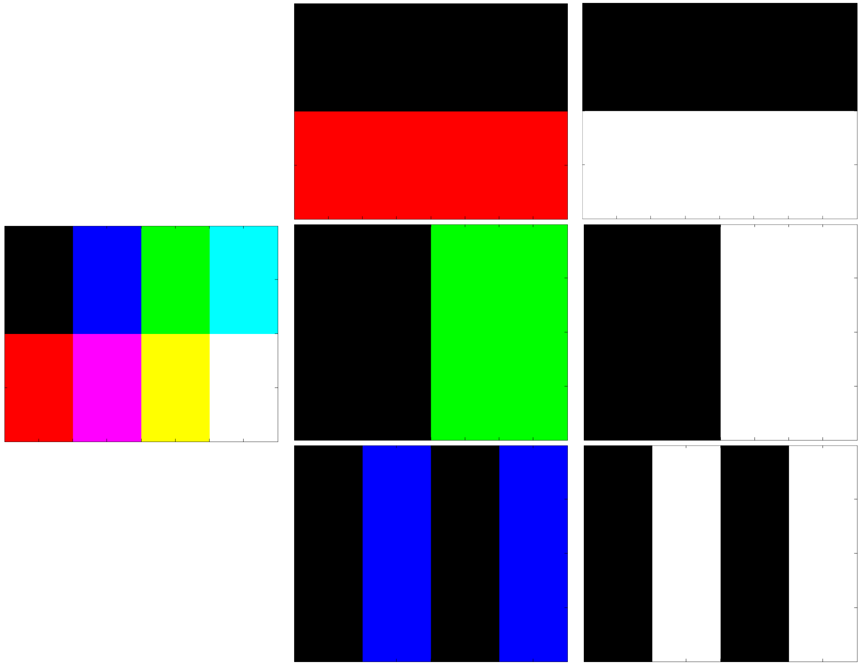


Color Spaces

- 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 (11/15/2009): 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:

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

RGB versus Grayscale

- RGB to grayscale conversion:

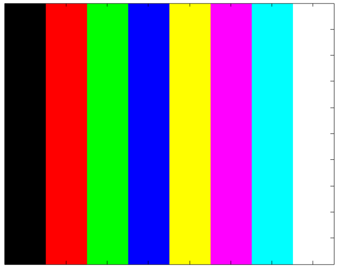
$$I(m,n) = 0.299R(m,n) + 0.587G(m,n) + 0.114B(m,n)$$

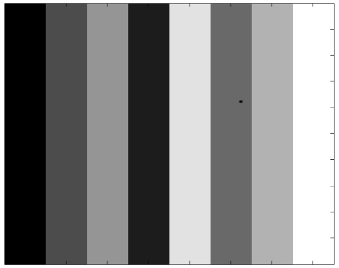
- Note: $0.299 + 0.587 + 0.114 = 1$.
- The luminance compensates for the eye's distinct sensitivity to different colors.
- The human eye is most sensitive to green, then red, and last blue.
  - There are evolutionary justifications for this difference.
  - A color with more green is brighter to the eye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

RGB versus Grayscale

- RGB to grayscale conversion:

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

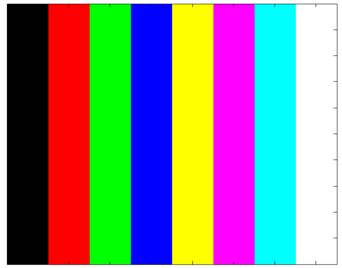




RGB versus Grayscale

- RGB to grayscale conversion:

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



$$\begin{aligned} \text{black} &= [0\ 0\ 0] = [R\ G\ B] \\ \text{red} &= [1\ 0\ 0] \\ \text{green} &= [0\ 1\ 0] \\ \text{blue} &= [0\ 0\ 1] \\ \text{yellow} &= [1\ 1\ 0] \\ \text{magenta} &= [1\ 0\ 1] \\ \text{cyan} &= [0\ 1\ 1] \\ \text{white} &= [1\ 1\ 1] \end{aligned}$$

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

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

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

Lowpass Filtering

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

Highpass Filtering

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

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

Highpass Filtering

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

Edge Enhancement

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

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

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

Edge Enhancement

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

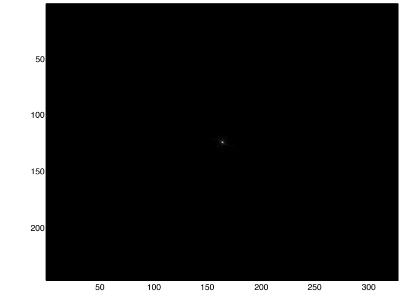
2-D Discrete Fourier Transform

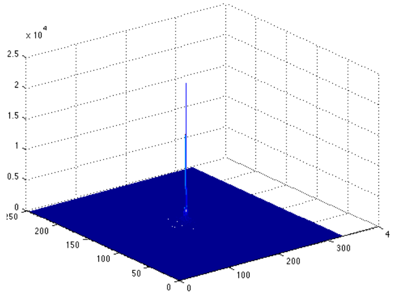
$$\mathcal{I}_F(U,V) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} I(m,n) e^{-j2\pi(Um+Vn)}$$

$I(m,n)$:

2-D Discrete Fourier Transform

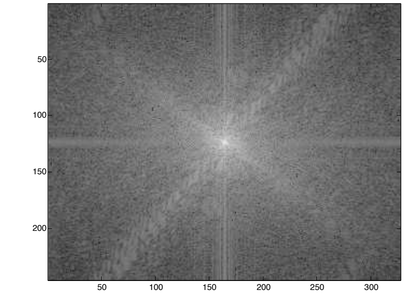
$\mathcal{I}_F(U, V)$:

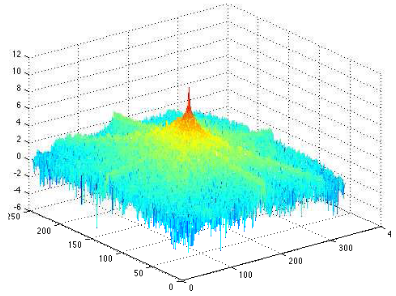




2-D Discrete Fourier Transform

$\mathcal{I}_F(U, V)$ on log-scale:





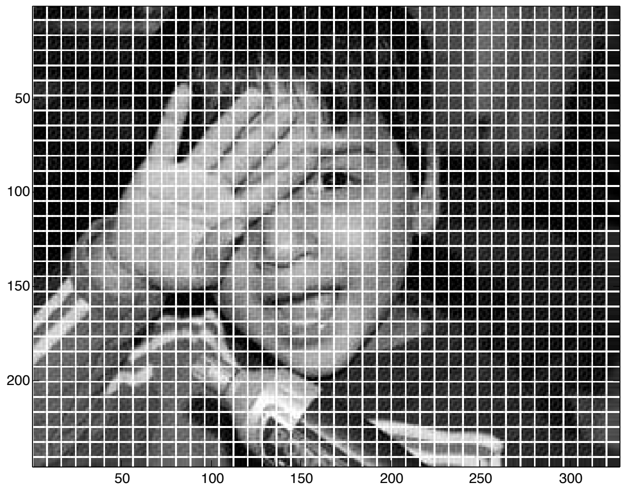
2-D Discrete Cosine Transform

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

$$\mathcal{I}_{DCT}(k, l) = \sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} I(m, n) \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right) k\right] \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right) l\right]$$

2-D Discrete Cosine Transform

2-D Discrete 8 × 8 Cosine Transform



2-D Discrete 8 × 8 Cosine Transform

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

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

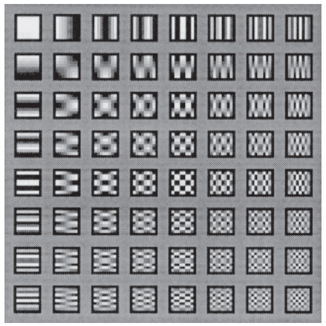
where

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{8}} & \text{for } k = 0 \\ \sqrt{\frac{2}{8}} & \text{for } k = 1, 2, \ldots, 7 \end{cases}$$

2-D Discrete 8 × 8 Cosine Transform

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

$$\cos\left[\frac{\pi}{8}\left(n+\frac{1}{2}\right)k\right]\cos\left[\frac{\pi}{8}\left(m+\frac{1}{2}\right)l\right] :$$



2-D Discrete 8 × 8 Cosine Transform

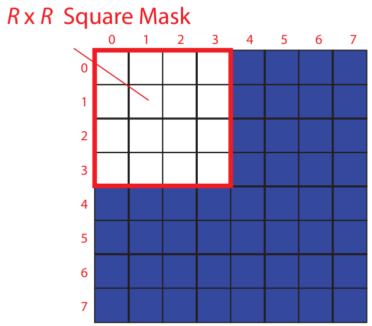
$$I^B(m,n) = \sum_{k=0}^{7}\sum_{l=0}^{7} \alpha(k)\alpha(l)\mathcal{I}^B_{DCT}(k,l)\cos\left[\frac{\pi}{8}\left(n+\frac{1}{2}\right)k\right]\cos\left[\frac{\pi}{8}\left(m+\frac{1}{2}\right)l\right]$$

Lossy 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

Consider removing (i.e., zeroing) signal components from 8×8-DCT domain outside a pre-defined mask.



Note: this is only an instructive example and there are multitudes of other ways to achieve this.

Lossy Compression via the DCT

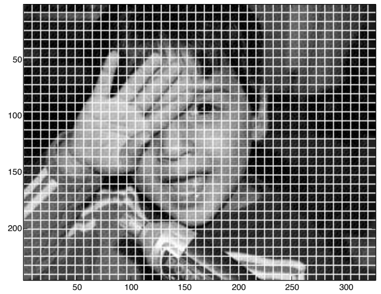
Step 1: Compute the 8×8-block DCT on $I(m, n)$.

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

Lossy Compression via the DCT

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

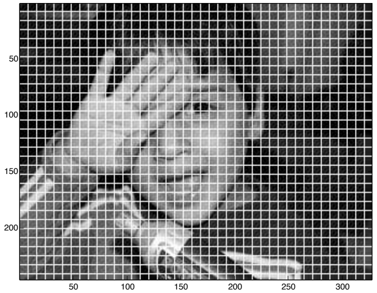
$I(m, n)$ and $I^{B}(m, n)$:

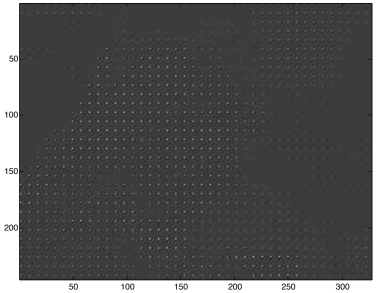


Lossy Compression via the DCT

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

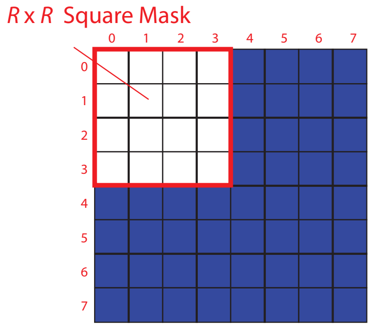
$I^B(m,n)$ and $\mathcal{I}^B_{DCT}(k,l)$:





Lossy Compression via the DCT

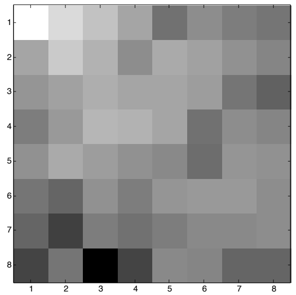
Step 2: Remove high-frequency components via $R \times R$ mask.

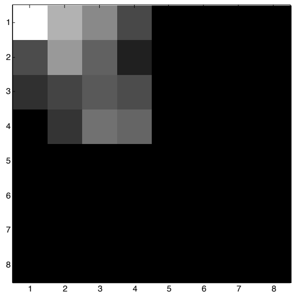


Lossy Compression via the DCT

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

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





Note: images displayed on log-amplitude scale.

Lossy Compression via the DCT

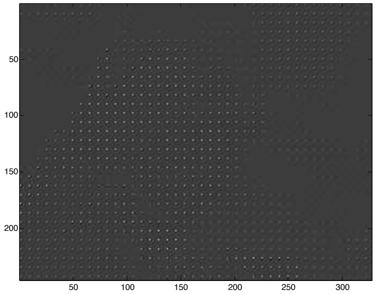
Step 3: Compute the 8×8-block IDCT on compressed DCT coefficients.

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

Lossy Compression via the DCT

Step 3: Compute the 8×8-block IDCT on compressed DCT coefficients.

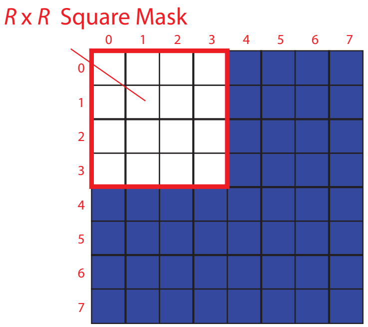
$\tilde{\mathcal{I}}^{B}_{DCT}(k,l)$ and $\tilde{I}(m,n)$ for $R = 4$:



Lossy Compression Results

Further Compression Gains

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



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