Midterm Review

CS 3570

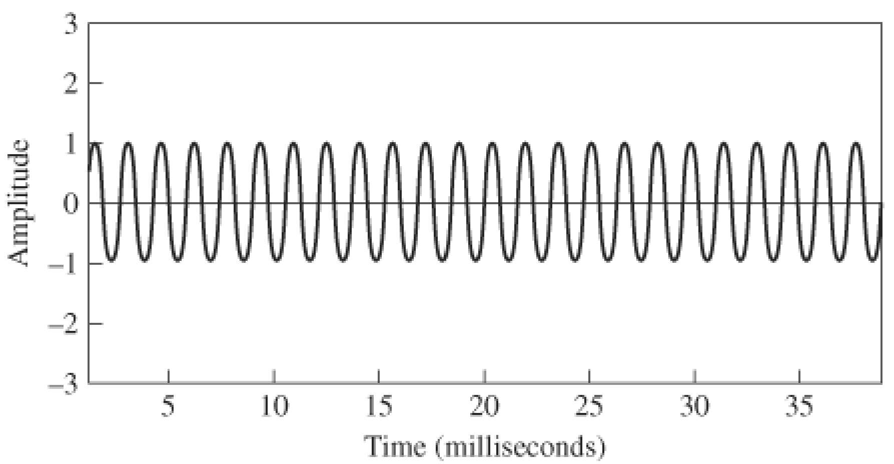
Unit 1. Digital Data Representation and Communication

A/D Conversion

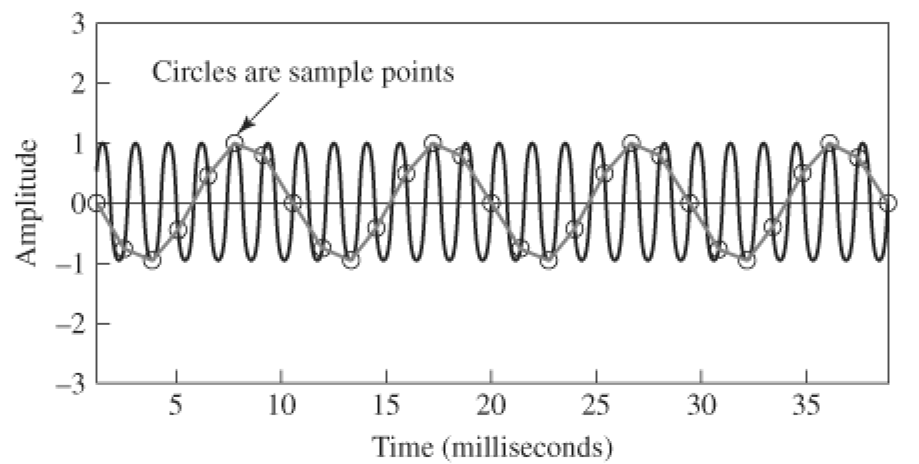
- Analog-to-digital conversion requires two steps: ***sampling*** and ***quantization***.
- The first step, sampling, chooses discrete points at which to measure a continuous phenomenon (***signal***).
- The number of samples taken per unit time or unit space is called the ***sampling rate*** or, alternatively, the ***resolution***.
- The second step, quantization, requires that each sample be represented in a fixed number of bits, called the ***bit depth***. The bit depth limits the precision with which each sample can be represented.

Sampling and Aliasing

- ***Undersampling:*** sampling rate does not keep up with the rate of change in the signal.
- ***Aliasing*** in a digital signal arises from undersampling and results in the sampled discrete signal cannot reconstruct the original source signal.



Audio wave at 637 Hz



637 Hz audio wave sampled at 770 Hz

Nyquist theorem

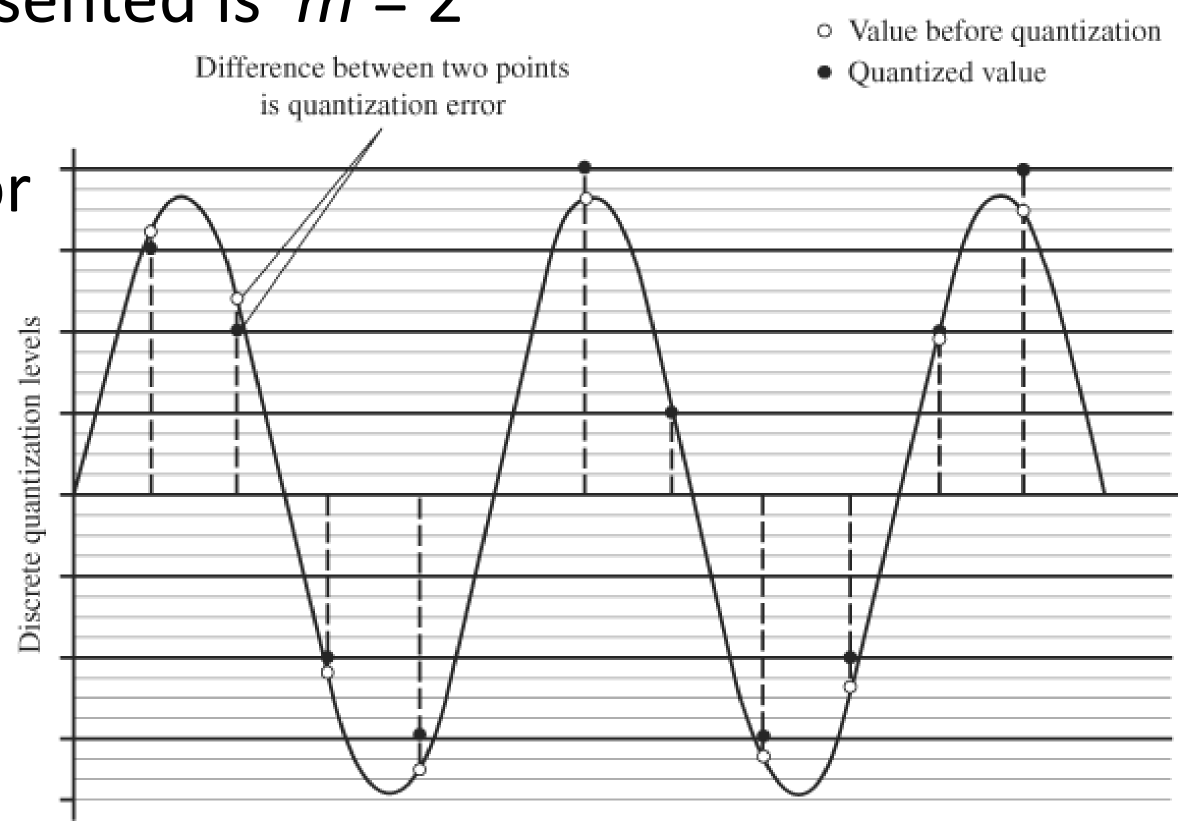
- The Nyquist theorem specifies the sampling rate needed for a given spatial or temporal frequency.
- To guarantee that no aliasing will occur, you must use a sampling rate that is greater than twice the maximal frequency in the signal being sampled.
- Let f be the frequency of a sine wave. Let r be the minimum sampling rate that can be used in the digitization process such that the resulting digitized wave is not aliased. Then the ***Nyquist frequency*** r is

$$r = 2f$$

Quantization

- Let n be the number of bits used to quantize a digital sample. Then the maximum number of different values that can be represented is $m = 2^n$

- Quantization error



Types of compression

- Lossless compression
  - No information is lost between the compression and decompression steps
  - Methods:
    - Run-Length encoding (RLE)
    - Entropy encoding
    - Arithmetic encoding

Types of compression

- Lossy compression
  - Sacrifice some information which is not important to human perception
    - In image files: subtle changes in color that the eye cannot detect
    - In sound files: changes in frequencies that are imperceptible to the human ear
  - Method:
    - Transform encoding

Entropy encoding

- Using fewer bits to encode symbols that occur more frequently, while using more bits for symbols that occur infrequently
- Shannon's equation, below, gives us a way a judging whether our choice of number of bits for different symbols is close to optimal.

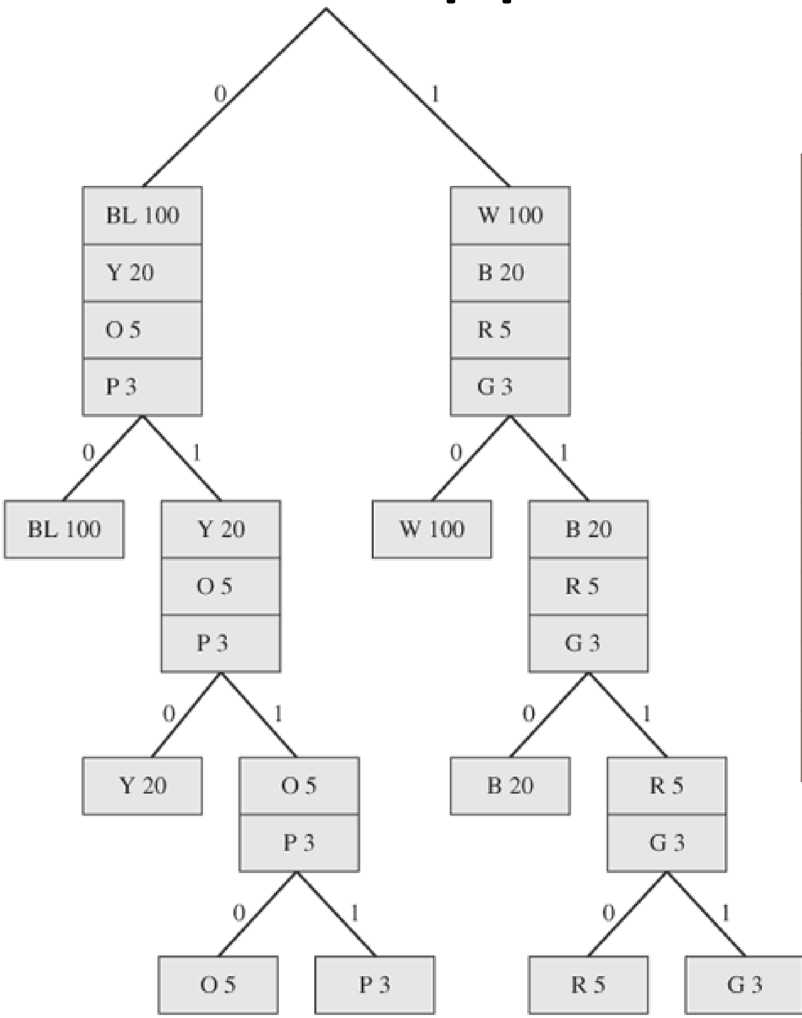
$$\cancel{H}(S) = \eta = \sum_i p_i log_2(\frac{1}{p_i})$$

  - *S* be a string of symbols and p_i be the frequency of the i^{th} symbol in the string.

Shannon-Fano algorithm

1. For a given list of symbols, develop a corresponding list of probabilities or frequency counts so that each symbol's relative frequency of occurrence is known.
2. Sort the lists of symbols according to frequency, with the most frequently occurring symbols at the left and the least common at the right.
3. Divide the list into two parts, with the total frequency counts of the left part being as close to the total of the right as possible.
4. The left part of the list is assigned to 0, and the right part is assigned to 1.
5. Recursively apply the steps 3 and 4 to each of the two halves, subdividing groups and adding bits to the codes until each symbol has become a corresponding code leaf on the tree.

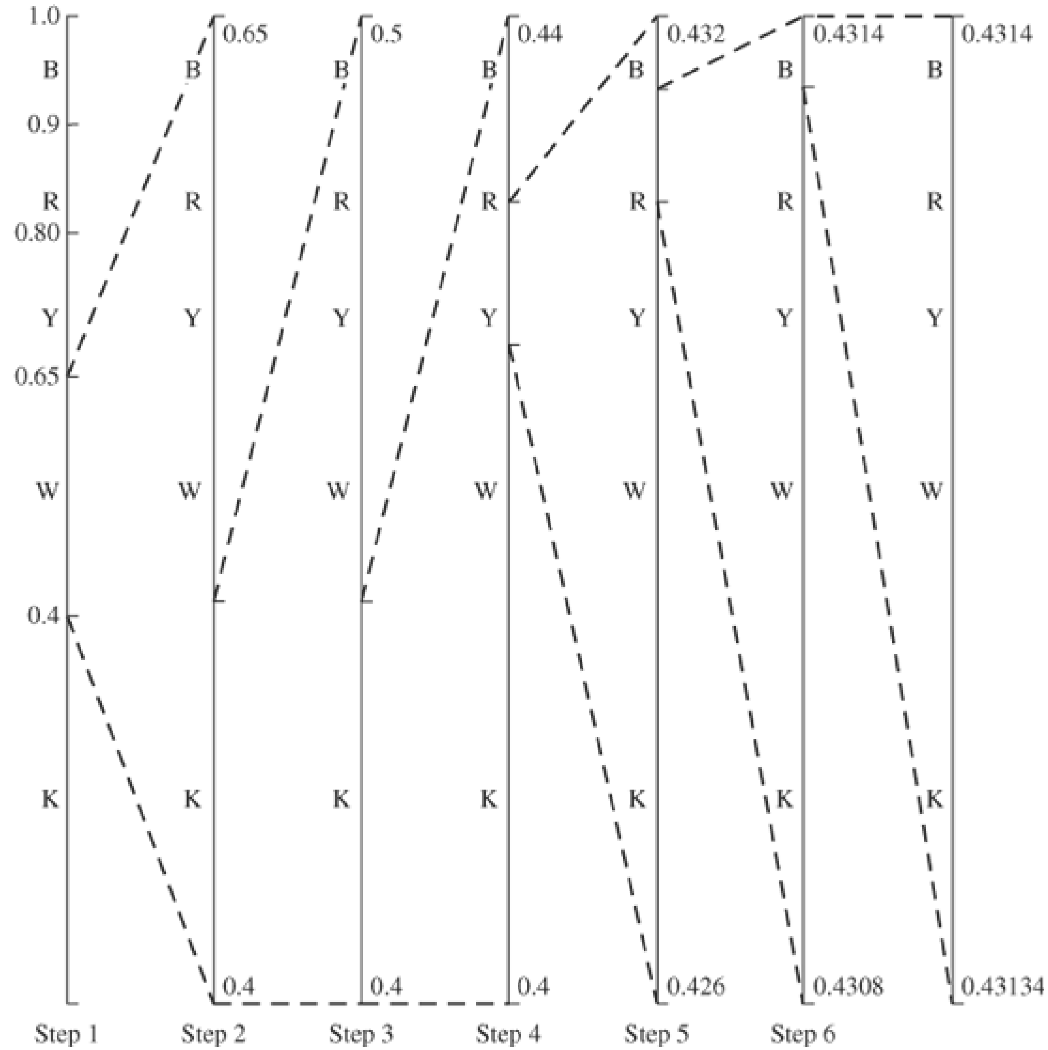
Result of the Shannon-Fano algorithm applied to compression



Color	Frequency	Code
black	100	00
white	100	10
yellow	20	010
orange	5	0110
red	5	1110
purple	3	0111
blue	20	110
green	3	1111

Arithmetic coding

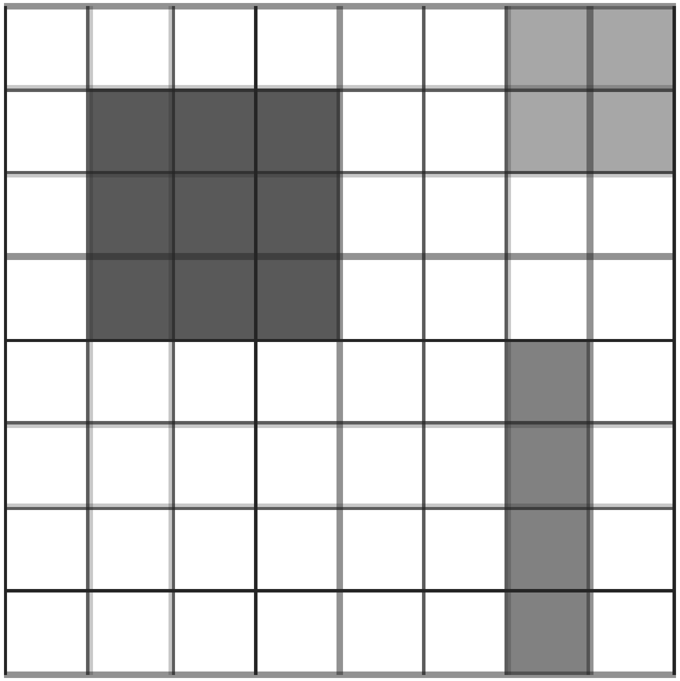
- Final encoding : 0.43137
  (0.43134+0.4314)/2=0.43137
- 0.43137 fits in the interval assigned to W -> W
- Remove the scaling of W
  (0.43137-0.4)/0.25=0.12548
  -> K
- Remove the scaling of K
  0.12548/0.4=0.3137
  -> K
- ……



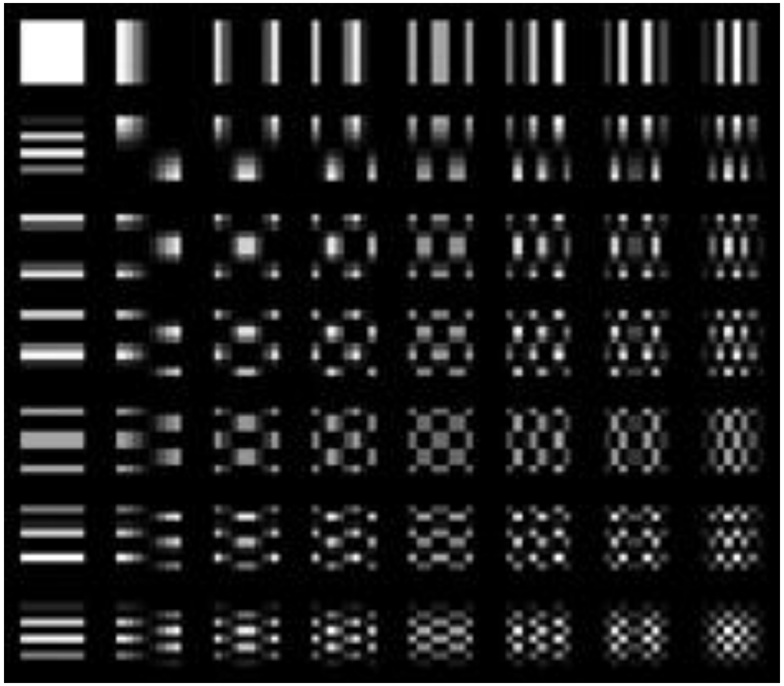
Transform Encoding

- Discrete cosine transform

Origin image (8x8)



Discrete cosine transform

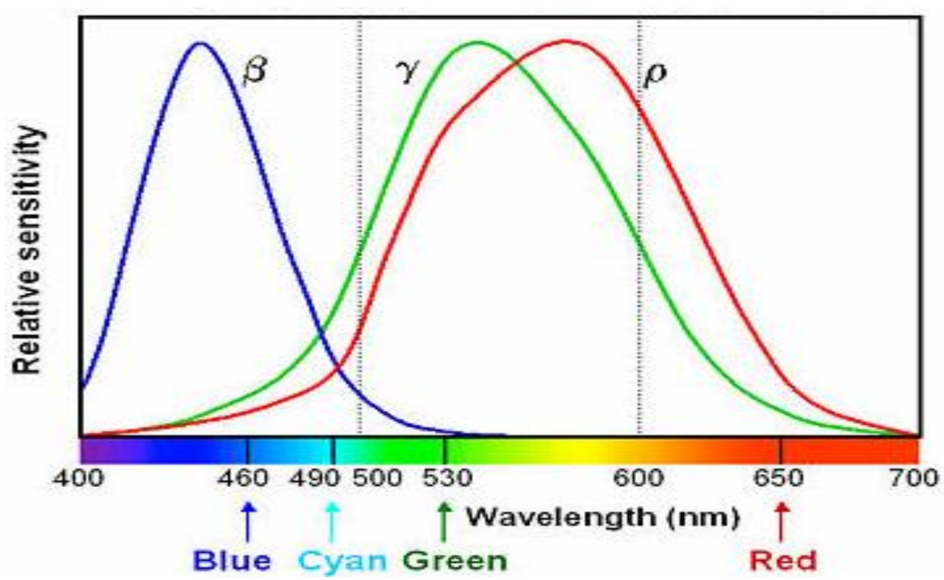


Unit 2. Digital Image Representation and Processing

CS 3570

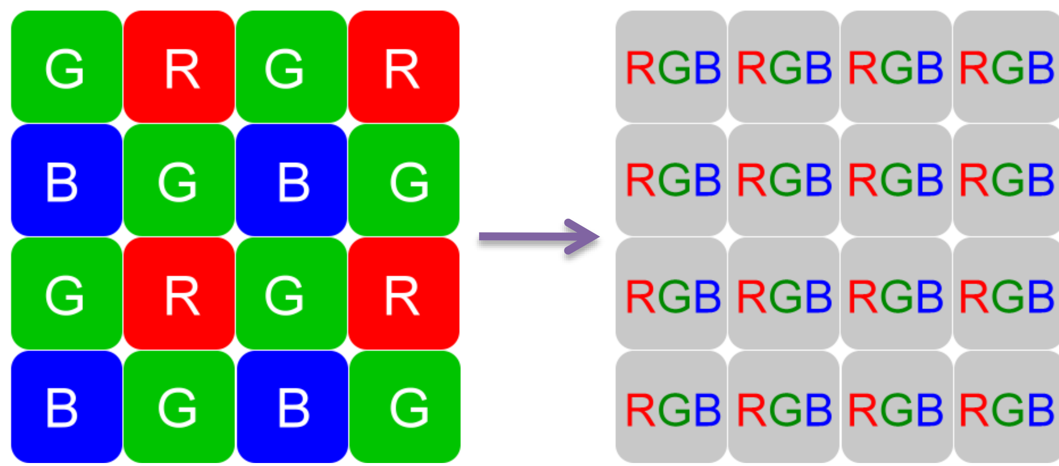
Color

- Color is composed of electromagnetic waves
  - The wavelength of visible color fall between approximately 370 and 780 nanometers.
- Three Characteristics of colors
  - Hue (essential color): *dominant wavelength*
  - Saturation (color purity)
  - Luminance (lightness)
- Color model
  - RGB color model
  - CMY color model
  - HSV color model
  - YIQ color model



Demosaicing

- The interpolation algorithm for deriving the two missing color channels at each photo-site is called ***demosaicing***
- Algorithm
  - Nearest neighbor
  - Linear
  - Cubic
  - Cubic spline



Dithering

- Thresholding
- Noise/random dithering
- Average dithering
- Pattern dithering
- Error Diffusion Dithering

Image transform

- Image transforms can be divided into two types
  - ***Pixel point processing***
    a pixel value is changed based only on its original value, without reference to surrounding pixels
  - ***Spatial filtering***
    changes a pixel’s value based on the values of neighboring pixels

Transform function

- In programs like Photoshop and GIMP, the Curves feature allows you to think of the changes you make to pixel values as a transform function
- We define a ***transform*** as a function that changes pixel values

$$g(x, y) = T(f(x, y)) \text{ , } p_2 = T(p_1)$$

  - $f(x, y)$ is the pixel value at that position (x, y) in the original image. Abbreviate $f(x,y)$ as p_1
  - T is the transformation function
  - $g(x, y)$ is the transformed pixel value. Abbreviate $f(x,y)$ as p_2

Histogram

- A ***histogram*** is a discrete function that describes frequency distribution; that is, it maps a range of discrete values to the number of instances of each value in a group of numbers.
- Let v_i be the number of instances of value i in the set of numbers. Let *min* be the minimum allowable value of i, and let *max* be the maximum. Then the ***histogram function*** is defined as

$$h(i) = v_i \text{ for } min \le i \le max.$$

Convolution

- Let $f(x, y)$ be an $M \times N$ image and $c(v, w)$ be an $m \times n$ mask. Then the equation for a linear convolution is

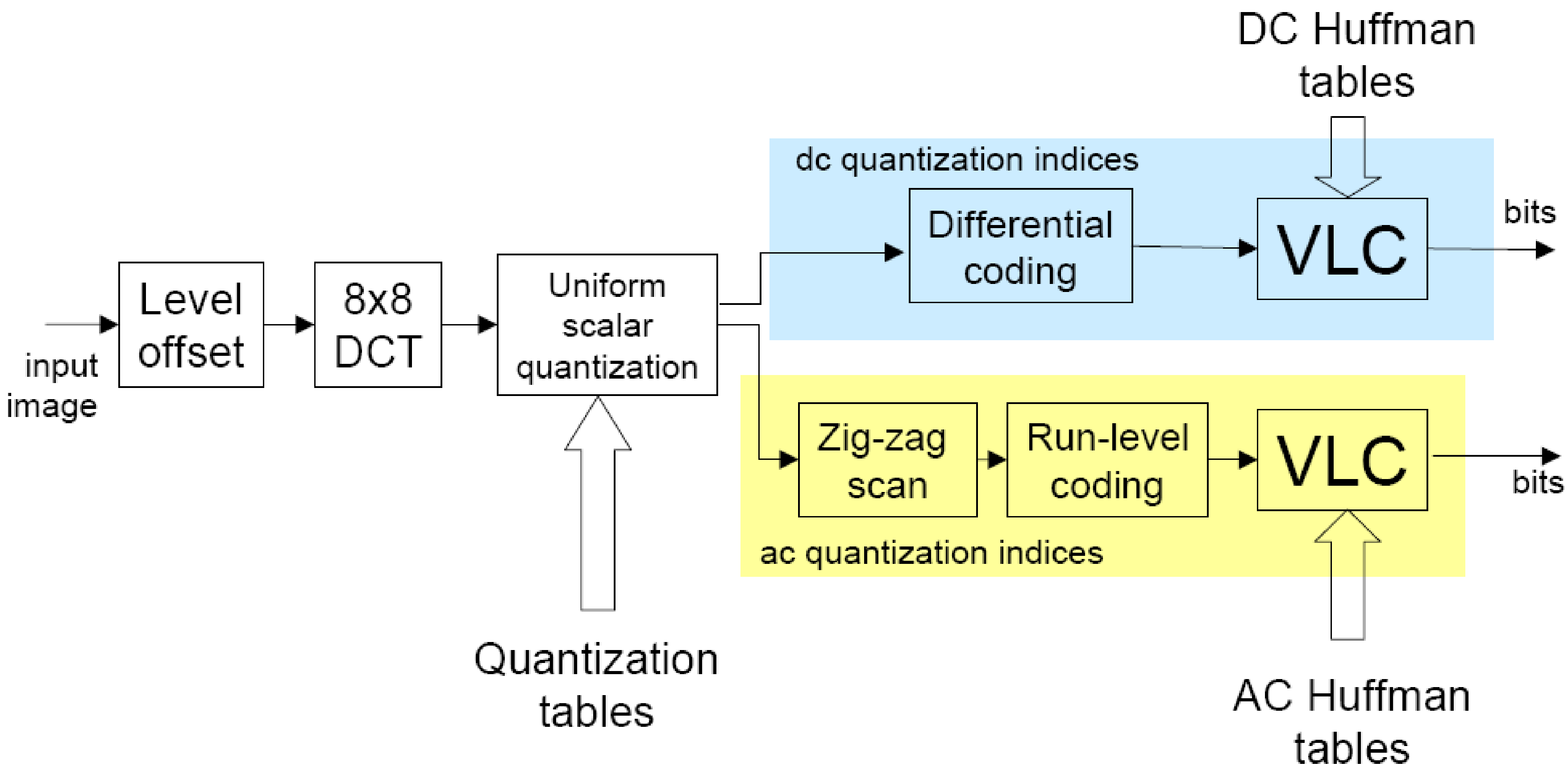
$$\mathrm{f}(x, y) = \sum_{v=-i}^{i} \sum_{w=-j}^{j} c(v, w) f(x - v, y - w)$$

where $i = (m - 1)/2$ and $j = (n - 1)/2$. Assume m and n are odd. This equation is applied to each pixel $f(x, y)$ of an image, for $0 \le x \le M - 1$ and $0 \le y \le N - 1$. (If $x - v < 0$, $x - v \ge M$, $y - w < 0$, or $y - w \ge N$, then $f(x, y)$ is undefined. These are edge cases, discussed below.)

Interpolation

- There are interpolation methods for resampling that give better results than simple replication or discarding of pixels
- ***Interpolation*** is a process of estimating the color of a pixel based on the colors of neighboring pixels
  - Nearest neighbor
  - Bilinear
  - Bicubic

JPEG Image Compression



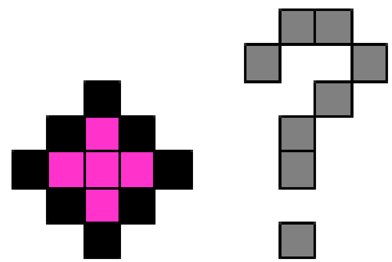
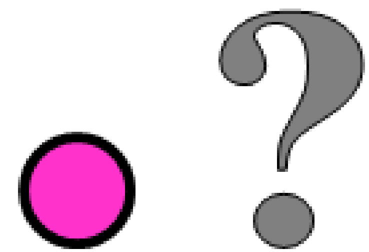
JPEG Compression Algorithm

```
algorithm jpeg
/*Input: A bitmap image in RGB mode.
Output: The same image, compressed.*/
{
  Divide image into 8 × 8 pixel blocks
  Convert image to a luminance/chrominance model such as YCbCr (optional)
  Shift pixel values by subtracting 128
  Use discrete cosine transform to transform the pixel data from the spatial domain
  to the frequency domain
  Quantize frequency values
  Store DC value (upper left corner) as the difference between current DC value and
  DC from previous block
  Arrange the block in a zigzag order
  Do run-length encoding
  Do entropy encoding (e.g., Huffman)
}
```

Unit 3
Graphics

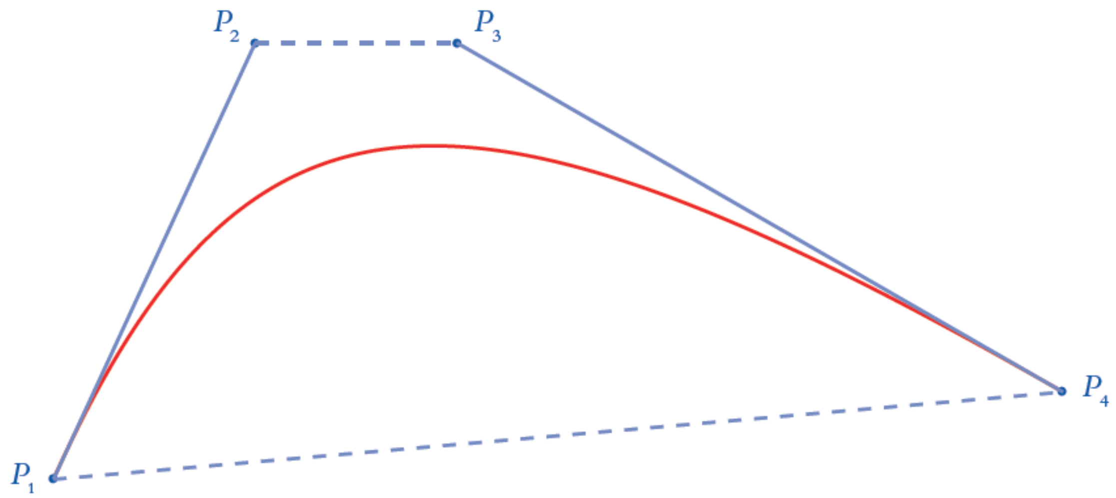
CS 3570

Bitmaps vs. Vector graphics

	Pixel	Vector
Example		
Grow and shrink	Bad	Good
Speed	Fast to create. Very hard to edit!	Slow to create. Much faster to edit.
Applications	Painter Photoshop	Power Point Illustrator

Cubic Bézier Curves

- Bézier curves of degree 3, commonly called "cubic Bézier curves", are the type most commonly found in vector graphics.
- They have just four control points: two are the **end points**, and the other two are called ***direction points.***



Bézier curve

- A Bézier curve is defined by a cubic polynomial equation
- ***P***(***t***) = ***T*** * ***M*** * ***G***
- $T = [t^3\ t^2\ t\ 1]$, ***M*** is called the ***basis matrix***. ***G*** is called the ***geometry matrix***.

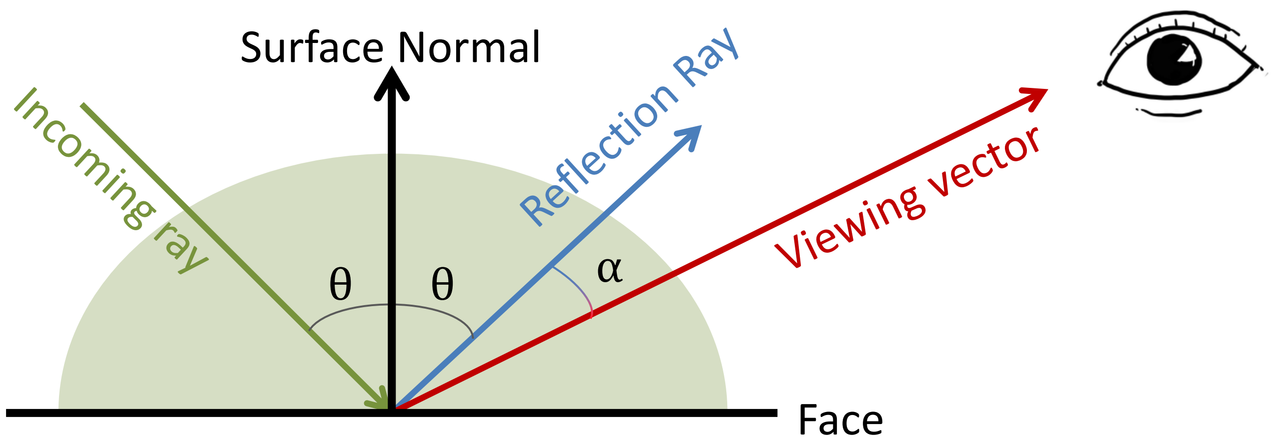
$$\boldsymbol{M} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{G} = \begin{bmatrix} \boldsymbol{p}_0 \\ \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix}$$

- The blending functions are given by $T * M$, $\boldsymbol{P}(t) = (\boldsymbol{T} * \boldsymbol{M}) * \boldsymbol{G} = (1 - \boldsymbol{t})^3 \boldsymbol{p}_0 + 3t(1 - t)^2 p_1 + 3t^2(1 - t)p_2 + t^3 p_3$
- Bézier curves can be described in terms of the blending functions

$$\boldsymbol{P}(t) = \sum_{k=0}^{n} \boldsymbol{p}_k\, blending_{k,n}(t) \quad \text{for} \quad 0 \le t \le 1$$

3D Graphics – Phong model

- Phong model
  - Intensity = Ambient + Diffuse + Specular
  - => $\underbrace{(I_a \times k_a)}_{\text{Ambient}} + \underbrace{(I_p \times k_d \times \cos\theta)}_{\text{Diffuse}} + \underbrace{(I_p \times k_s \times (\cos\alpha)^n)}_{\text{Specular}}$



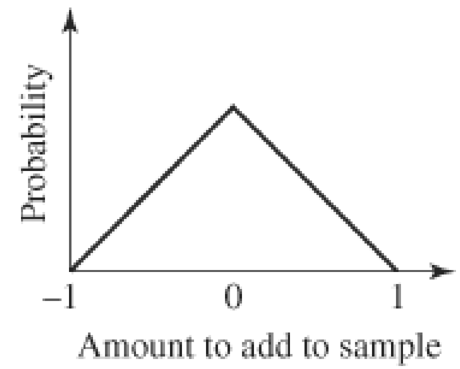
Unit 4. Digital Audio Representation and Processing

CS 3570

Audio Dithering

- Dithering function
  - Triangular probability density function (TPDF)

Triangular probability function



  - Rectangular probability density function (RPDF): All numbers in the selected range have the same probability
  - Gaussian PDF: The Gaussian PDF weights the probabilities according to a Gaussian
  - Colored dithering: Colored dithering produces noise that is not random and is primarily in higher frequencies.

Discrete Fourier Transform

- The ***discrete Fourier transform*** (**DFT**) operates on an array of *N* audio samples, returning cosine and sine coefficients that represent the audio data in the frequency domain.

$$F_n = \frac{1}{N}\sum_{k=0}^{N-1} f_k \cos\left(\frac{2\pi nk}{N}\right) - i f_k \sin\left(\frac{2\pi nk}{N}\right) \qquad (4.8)$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} f_k e^{\frac{-i2\pi nk}{N}}$$

Discrete Fourier Transform

- Let $\boldsymbol{f}_k$ be a discrete integer function representing a digitized audio signal in the time domain, and $\boldsymbol{F}_n$ be a discrete, complex number function representing a digital audio signal in the frequency domain. Then the ***inverse discrete Fourier transform*** is defined by

$$\boldsymbol{f}_k = \sum_{n=0}^{N-1}\left[\boldsymbol{a}_n \cos\left(\frac{2\pi nk}{N}\right) + \boldsymbol{b}_n \sin\left(\frac{2\pi nk}{N}\right)\right] \qquad (4.7)$$

$$= \sum_{n=0}^{N-1} F_n e^{\frac{i2\pi nk}{N}}$$

- Subscript k: signal value at time k

  Subscript n: n_{th} frequency component

Windowing Function

- **Window function** - to reduce the amplitude of the sound wave at the beginning and end of the FFT window. If the amplitude of the wave is smaller at the beginning and end of the window, then the spurious frequencies will be smaller in magnitude as well.

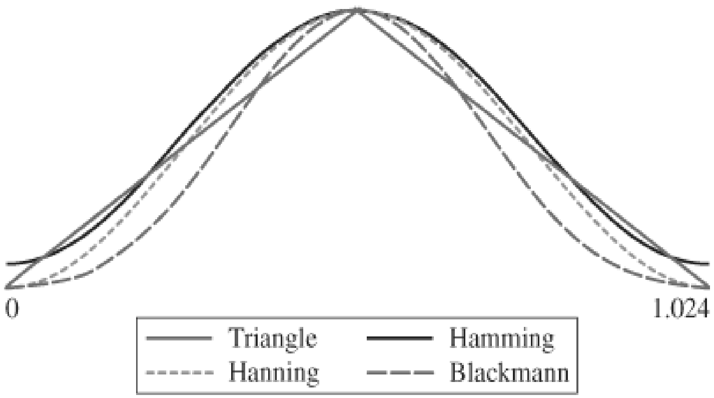
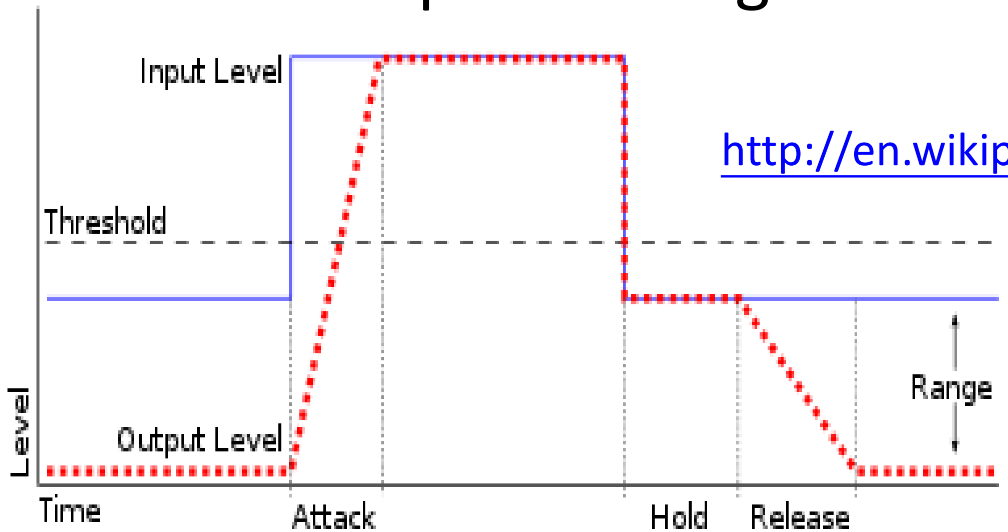


TABLE 4.4 Windowing Function for FFT

$u(t) = \begin{cases} \frac{2t}{T} & \text{for } 0 \le t < \frac{T}{2} \\ 2 - \frac{2t}{T} & \text{for } \frac{T}{2} \le t \le T \end{cases}$ Triangular windowing function	$u(t) = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi t}{T}\right)\right] \quad \text{for} \quad 0 \le t \le T$ Hanning windowing function
$u(t) = 0.54 - 0.46\cos\left(\frac{2\pi t}{T}\right) \quad \text{for} \quad 0 \le t \le T$ Hamming windowing function	$u(t) = 0.42 - 0.5\cos\left(\frac{2\pi t}{T}\right) + 0.08\cos\left(\frac{4\pi t}{T}\right)$ for $0 \le t \le T$ Blackman windowing function

Noise Gating

- A **noise gate** allows a signal to pass through only when it is above a set threshold.
- It is used when the level of the signal is above the level of the noise. It does not remove noise from the signal. When the gate is open, both the signal and the noise will pass through.



http://en.wikipedia.org/wiki/Noise_gate

FIR Filter

- Let $x(n)$ be an audio signal of L samples for $0 \le n \le L - 1$
  y(n) be the filtered signal
  h(n) be the convolution mask
- The FIR filter function is defined by

$$y(n) = h(n) \otimes x(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

$$where\ x(n-k) = 0\ if\ n-k < 0$$

- The number of the coefficients of h is the **order of the filter.**

IIR Filter

- The recursive form of IIR filter is

$$y(n) = h(n) \otimes x(n) = \sum_{k=0}^{N-1} a_k x(n-k) - \sum_{k=1}^{M} b_k y(n-k)$$

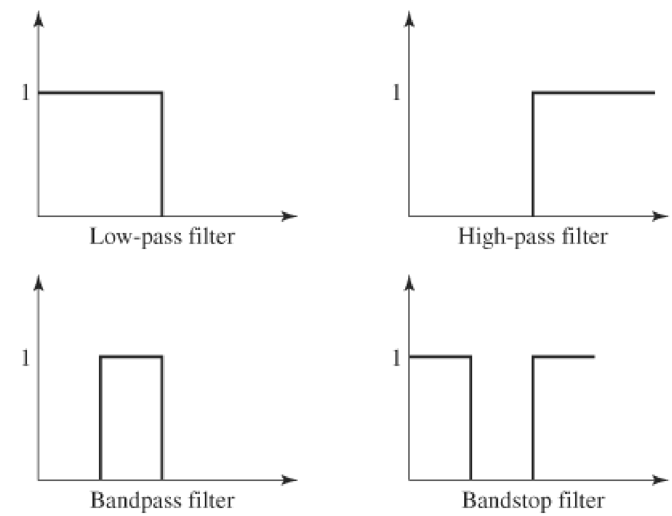
a_k is the coefficient of the forward filter

b_k is the coefficient of the feedback filter

- y(n) depends on present and past input samples as well as on past outputs.

Filters in Audio Processing

- **Band filters**
  - **low-pass filter**—retains only frequencies below a given level.
  - **high-pass filter**—retains only frequencies above a given level.
  - **bandpass filter**—retains only frequencies within a given band.
  - **bandstop filter**—eliminates all frequencies within a given band.



Horizontal axis: Frequency component
Vertical axis: Fraction of frequency component retained in filtered signal

FIR Filter Design – Ideal Filter

TABLE 5.2 Equations for Ideal Impulse Responses for Standard Filters, Based on Cutoff Frequency f_c and Band Edge Frequencies f_1 and f_2

Type of filter	$h_{ideal}(n),\ n \neq 0$	$h_{ideal}(0)$
Low-pass	$\dfrac{\sin(2\pi f_c n)}{\pi n}$	$2f_c$
High-pass	$-\dfrac{\sin(2\pi f_c n)}{\pi n}$	$1 - 2f_c$
Bandpass	$f_2 \dfrac{\sin(2\pi f_2 n)}{\pi n} - f_1 \dfrac{\sin(2\pi f_1 n)}{\pi n}$	$2(f_2 - f_1)$
Bandstop	$f_1 \dfrac{\sin(2\pi f_1 n)}{\pi n} - f_2 \dfrac{\sin(2\pi f_2 n)}{\pi n}$	$1 - 2(f_2 - f_1)$

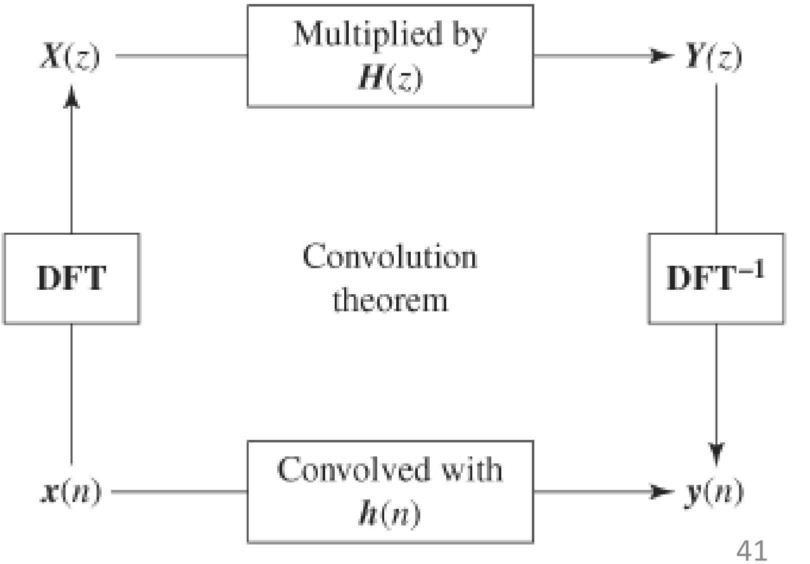
FIR Filter Design – Ideal to Real

- A sinc function goes on infinitely in the positive and negative directions … how can we implement an FIR filter?
- One way is to multiply the ideal impulse response by a **windowing function**. The purpose of the windowing function is to make the impulse response finite.
- However, making the impulse response finite results in a frequency response that is less than ideal, containing “ripples”.

Relationship Between Convolution and Fourier Transform

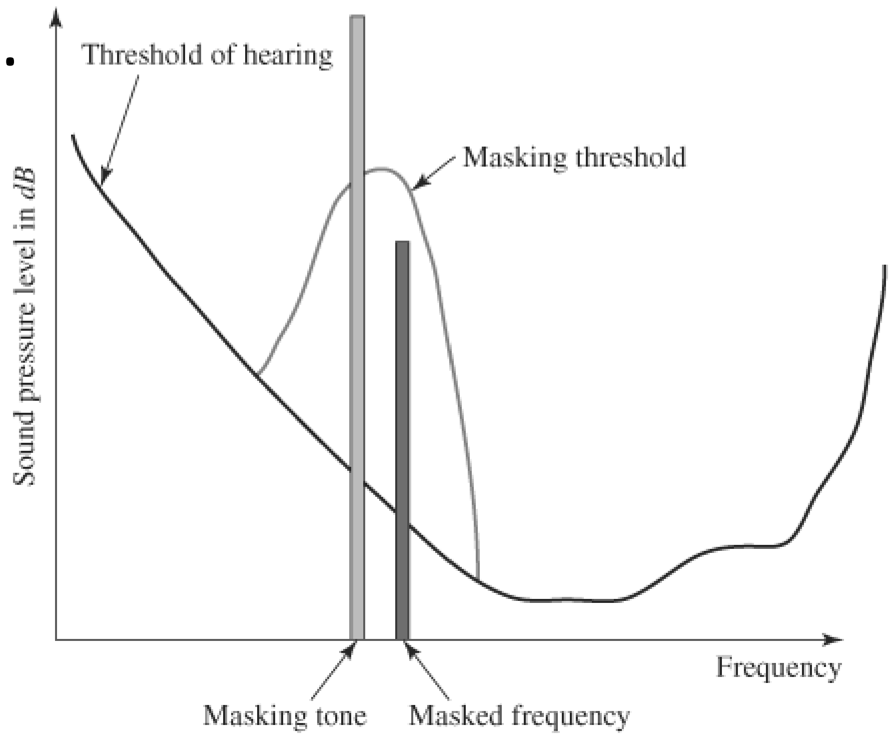
- Let $\boldsymbol{H}(z)$ be the discrete Fourier transform of a convolution filter $\boldsymbol{h}(n)$, and let $\boldsymbol{X}(z)$ be the discrete Fourier transform of a digital audio signal $\boldsymbol{x}(n)$. Then $y(n) = h(n) \otimes x(n)$ is equivalent to the inverse discrete Fourier transform of $\boldsymbol{Y}(z)$, where $\boldsymbol{Y}(z) = \boldsymbol{H}(z)\boldsymbol{X}(z)$.

Equivalent operations in time and frequency domains



Psychoacoustics

- **Masking** - Within a small window of time, the loudest frequency sound can overpower the others in the critical band, making human unable to hear the other frequencies.



MPEG-1 Audio Compression

ALGORITHM 5.2

```
algorithm MPEG-1_audio
/*Input: An audio file in the time domain
  Output: The same audio file, compressed*/
{
  Divide the audio file into frames
  For each frame {
    By applying a bank of filters, separate the signal into frequency bands.
    For each frequency band {
      Perform a Fourier transform to analyze the band's frequency spectrum
      Analyze the influence of tonal and nontonal elements (i.e., transients)
      Analyze how much the frequency band is influenced by neighboring bands
      Find the masking threshold and signal-to-mask ratio (SMR) for the band,
      and determine the bit depth in the band accordingly
      Quantize the samples from the band using the determined bit depth
      Apply Huffman encoding (optional)
    }
    Create a frame with a header and encoded samples from all bands
  }
}
```