Introduction to Computer Science & Engineering

Lecture 3: Data Representation and Storing

Jeonghun Park

KYUNGPOOK NATIONAL UNIVERSITY

Data and Computers

- Computers are multimedia device, which deals with a vast range of information
  - Numbers, Text, Audio, Video,..
- All of them should be stored into **bits**

jeonghun.park@knu.ac.kr

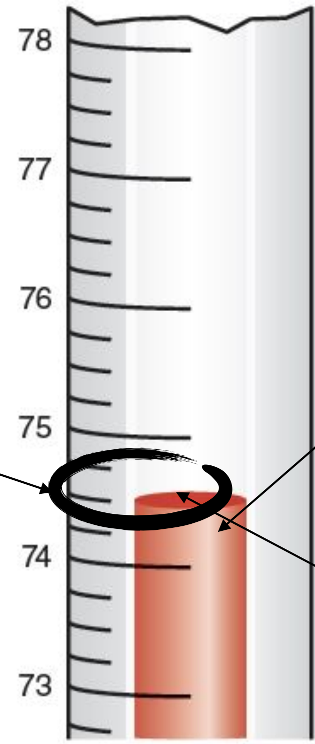
Analog and Digital Information

- What is the fundamental difference?
    - **Analog**: A continuous data, analogous to the actual information (Dimension is infinite)
    - **Digital**: A discrete data, breaking the information up into separate elements (Dimension is finite)

jeonghun.park@knu.ac.kr

Analog and Digital Information

A value recorded by markers
= Digital



The amount of mercury
= Analog

Quantization error
= Loss of information during ***digitize***

FIGURE 3.1 A mercury thermometer continually rises in direct proportion to the temperature

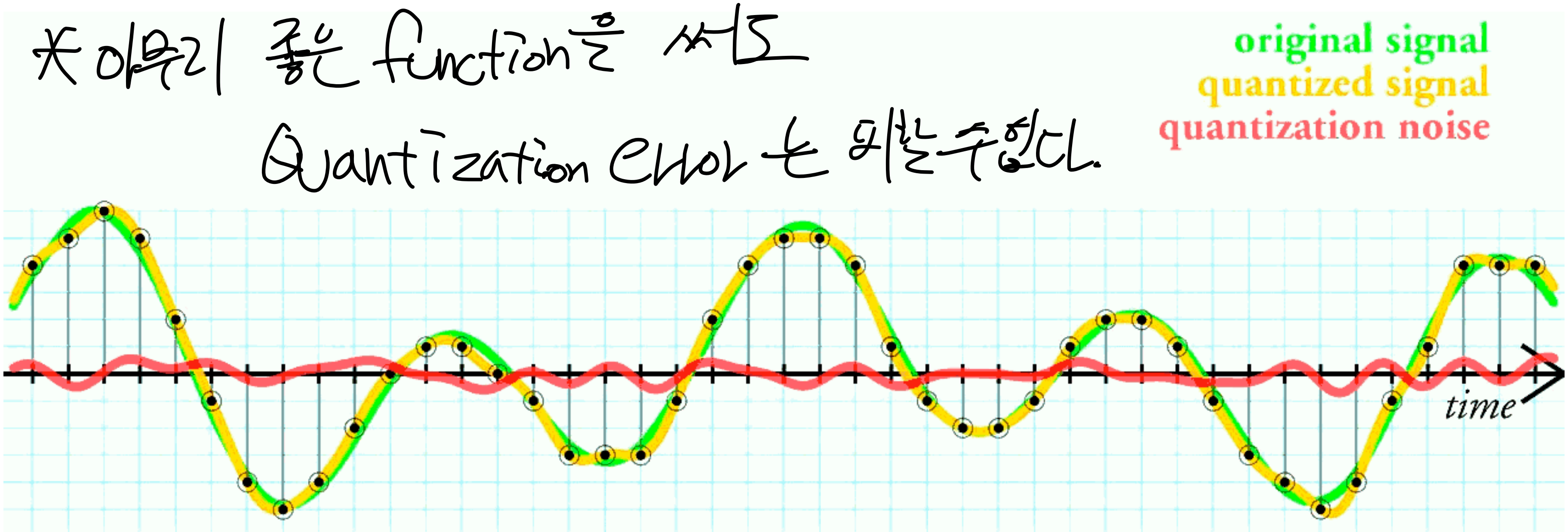
①아날로그 → 디지털 변환시 생기는 오류

②진수 변환시 생기는 오류

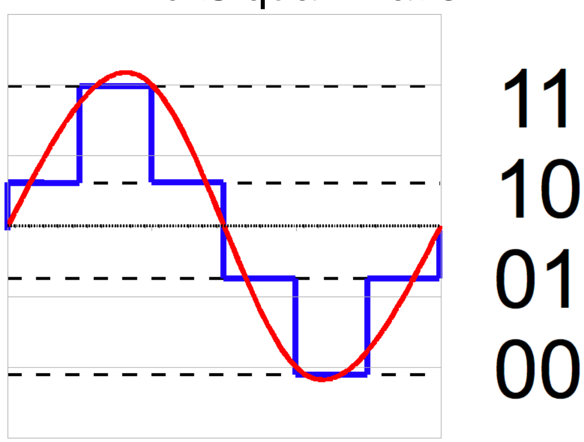
jeonghun.park@knu.ac.kr

Quantization (= Digitization)

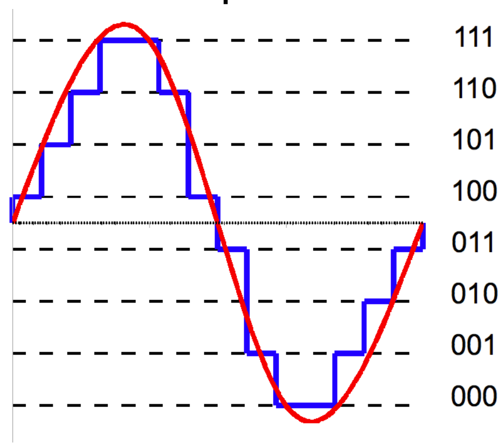
* 아무리 좋은 function을 써도
Quantization error는 피할수없다.



2 bits quantization



3 bits quantization



jeonghun.park@knu.ac.kr

Error Effects

FIGURE 3.2 An analog signal and a digital signal

손상발생시 복구불가

손상발생해도 복구가능. *단 심한 손상은 [illegible], 복구불가

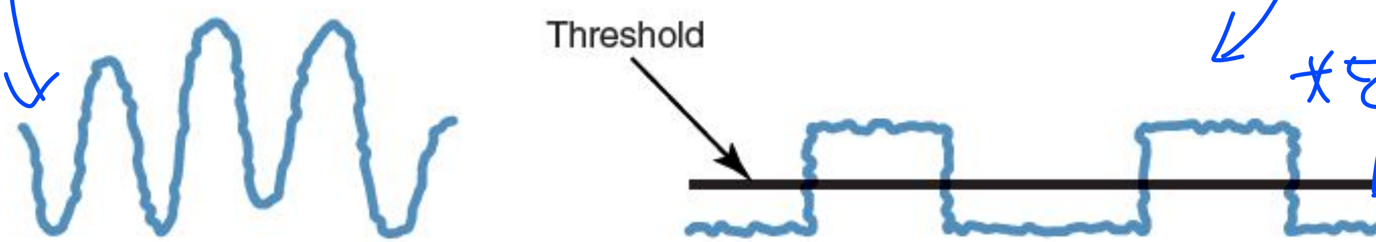


FIGURE 3.3 Degradation of analog and digital signals

Digital 신호는 작은 에러는 양호하지만 큰 에러는 치명적

- Every signal suffers from error caused by various sources
- Digital signals are robust with small error, but large error can be more fatal **(Why?)**

jeonghun.park@knu.ac.kr

Binary Representations

1 Bit	2 Bits	3 Bits	4 Bits	5 Bits
0	00	000	0000	00000
1	01	001	0001	00001
	10	010	0010	00010
	11	011	0011	00011
		100	0100	00100
		101	0101	00101
		110	0110	00110
		111	0111	00111
			1000	01000
			1001	01001
			1010	01010
			1011	01011
			1100	01100
			1101	01101
			1110	01110
			1111	01111
				10000
				10001
				10010
				10011
				10100
				10101
				10110
				10111
				11000
				11001
				11010
				11011
				11100
				11101
				11110
				11111

FIGURE 3.4 Bit combinations

The total # of things can be represented with n bits = 2^n

jeonghun.park@knu.ac.kr

Some Notes

- How many bits are needed to represent 10 kinds of information?
  - $2^3 < 10 < 2^4$
  - 4 bits are enough
- How many bits are needed to represent 30 kinds of information?
  - $2^4 < 30 < 2^5$
  - 5 bits are enough
- How about x kinds of information?

jeonghun.park@knu.ac.kr

S16.XX Revisit

- For simplicity, let's consider S16.00 format
  - Maximum value: $7\text{FFF} = 0111111111111111$
    $$= 2^{14} + 2^{13} + ... + 2^{0}$$
    $$= 2^{15} - 1 = 32767$$
  - Minimum value: $8000 = 1000000000000000$
    $$= -2^{1}5 = -32768$$

jeonghun.park@knu.ac.kr

Insights behind S16.XX

- Why we do this?
  - Consider a more simple way:
    - Let the leading bit directly indicates the sign
  - Example) $\mathrm{FFFF} = 1111111111111111$
    $= -111111111111111$
    $= -32767$
  - Duplicated zero problem
    - $0000 = 8000$
    - Waste of memory, since two different symbols are used for the same information

Insights behind S16.XX (Contd.)

- A solution
  - Divide and assign
  - For example, if we have a feasible range from 0 to 15, use 0,…,7 for positive numbers and use 8,…,15 for negative numbers
  - With this way, only one symbol is assigned to zero so we can save the resources
- This is exactly S16.XX format does
  - **How?**

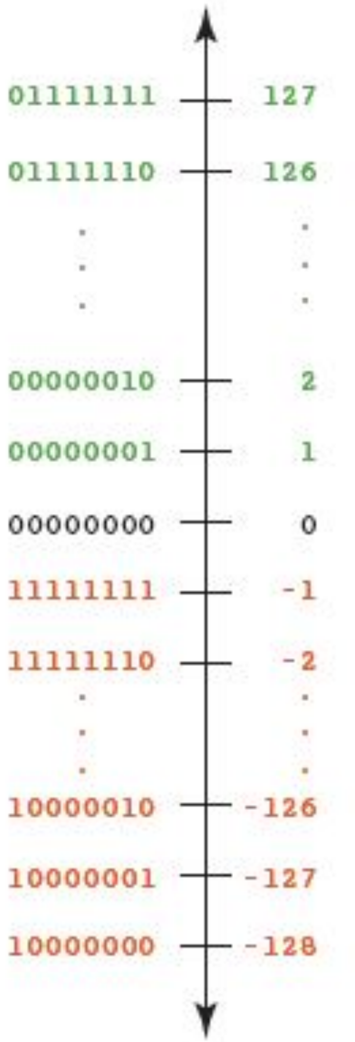
jeonghun.park@knu.ac.kr

Two's Complement

- Given an arbitrary number x, its negative value is represented as
  - $N(x) = 2^{16} - x$ # of bits
- For example, considering x = 1000, its two's complement is
  - $N(1000) = 2^{16} - 1000 = \mathrm{FC18}$
- FC18 = 1111110000011000
  - Following the computation rule, its value is $2^3 + 2^4 + 2^{10} + 2^{11} + 2^{12} + 2^{13} + 2^{14} - 2^{15} = -1000$

jeonghun.park@knu.ac.kr

Some Notes



- Intuitive vertical representation of a binary system

jeonghun.park@knu.ac.kr

Arithmetics on S16.XX

- Assume that we let 0,…,49 represent 0,…,49 and let 50,…,99 represent -50,…,-1
  - Then the addition can be obtained as follows

Signed-Magnitude	New Scheme
$\begin{array}{r} 5 \\ + -6 \\ \hline -1 \end{array}$	$\begin{array}{r} 5 \\ + \ 94 \\ \hline 99 \end{array} \longrightarrow -1$
$\begin{array}{r} -4 \\ + \ 6 \\ \hline 2 \end{array}$	$\begin{array}{r} 96 \\ + \ 6 \\ \hline 2 \end{array}$
$\begin{array}{r} -2 \\ + -4 \\ \hline -6 \end{array}$	$\begin{array}{r} 98 \\ + \ 96 \\ \hline 94 \end{array} \longrightarrow -6$

jeonghun.park@knu.ac.kr

Arithmetics on S16.XX

- To perform subtraction x - y, we first obtain (-y) in our representation system and calculate

Signed-Magnitude	New Scheme	Add Negative
− 5	95	95
− 3	− 3	+ 97
− 8		92

jeonghun.park@knu.ac.kr

Arithmetics on S16.XX (Contd.)

- With 16 bits, our original range is 0,…, 65535
  - We use 0,…, 32767 for positive numbers
  - We use 32768,…,65535 for negative numbers
- So our arithmetics can be done as follows
  - 1234 + (-1243) = -9

$\Leftrightarrow$ 1234 + 64293 = 65527 $\Rightarrow$ **-9**

jeonghun.park@knu.ac.kr

Overflow

- What if a summation result is larger than the maximum value?
  - ```
       01111111
    +  00000011
    -----------
       10000010
    ```

Summation of two positive numbers = Negative?

- In S16.00, this happens when the summation result is larger than 32767

jeonghun.park@knu.ac.kr
    ```

How to Avoid Overflow

- Check-before-Go
  - We should check the summation result is larger than the maximum value that can be represented in the corresponding systems
  - If the summation result is smaller?
    - Do what we do
  - If the summation result is larger?
    - A special mapping is needed

jeonghun.park@knu.ac.kr

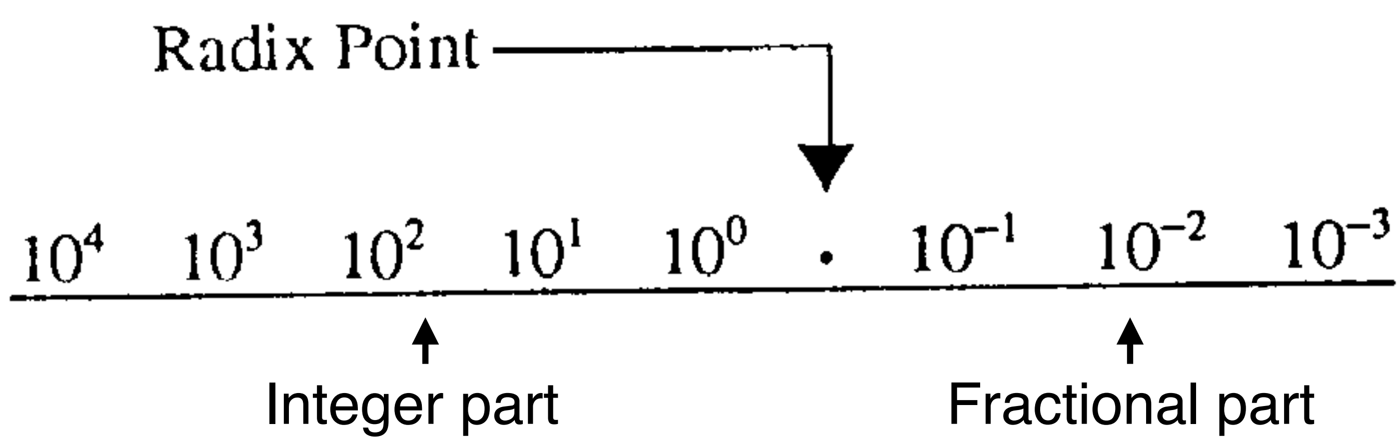
Max-Min Function

- Max function
  - Max(a, b) = a if a > b, or Max(a, b) = b otherwise
- Min function
  - Min(a, b) = a if a < b, or Min(a, b) = b otherwise
- Let M = the maximum value and m = minimum value, and we compute a + b
  - Then we do Min(Max((a+b), m),M)
  - This prevents overflow

jeonghun.park@knu.ac.kr

Radix Point

- Any number can be separated into
  - Integer part (a value larger than 1)
  - Fractional part (a value smaller than 1)
- A radix point is a separation point between them
- For example, in decimal number systems,



jeonghun.park@knu.ac.kr

Radix Point (Contd.)

- For example, in a S16.15 format:
  - 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

    Fractional part

    2^{-1}

- For example, in a S16.10 format:
  - 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

    Integer part

    Fractional part

    2^0 2^{-1}

Then where is the radix point?

jeonghun.park@knu.ac.kr

Fixed and Floating Point

- A radix point of S16.XX is fixed
  - Fixed point
- Floating point
  - We can flexibly set a radix point
  - A decimal number system we generally use has a floating number
- A computer generally uses fixed point numbers for a complexity issue

jeonghun.park@knu.ac.kr

Mantissa-Exponent

exponent part

S16.xx | 0000

00000 이면 S16.X3 써

- S16.XX format has a fixed point

0001 이면 S16.1X2 써 등

- What is a limitation of a fixed point system?
    - Its coverable range is fixed

약속할 수 있음

    - If inputs has a large dynamic range, a fixed point system could have a large error

(소수점이동)

- How to overcome this?
    - Mantissa-Exponent: Floating-point-like operation in computers

$$\text{sign} * \text{mantissa} * 10^{\text{exp}}$$

jeonghun.park@knu.ac.kr

Text Data Compression

양자화 infinite → finite 로 바꿈

- The number of characters to represent is ***finite***, so list them all and assign each a binary string
- A list of characters and the codes used to represent each one
- Computer manufacturers agreed to standardize

jeonghun.park@knu.ac.kr

The ASCII Character Set

7bits (아스키) (미국 내에선 괜찮음)

- **ASCII** stands for **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
- **ASCII** originally used seven bits to represent each character, allowing for 128 unique characters
- Later extended **ASCII** evolved so that all eight bits were used

jeonghun.park@knu.ac.kr

ASCII Map

Left Digit(s) / Right Digit	ASCII 0	1	2	3	4	5	6	7	8	9
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT
1	LF	VT	FF	CR	SO	SI	DLE	DC1	DC2	DC3
2	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS
3	RS	US	□	!	"	#	$	%	&	'
4	(	)	*	+	,	-	.	/	0	1
5	2	3	4	5	6	7	8	9	:	;
6	<	=	>	?	@	A	B	C	D	E
7	F	G	H	I	J	K	L	M	N	O
8	P	Q	R	S	T	U	V	W	X	Y
9	Z	[	\	]	^	_	`	a	b	c
10	d	e	f	g	h	i	j	k	l	m
11	n	o	p	q	r	s	t	u	v	w
12	x	y	z	{	|	}	~	DEL		

FIGURE 3.5 The ASCII character set

jeonghun.park@knu.ac.kr

The Unicode Character Set

이걸로도 부족해서 유니코드 사용.

- **Extended ASCII** is not enough for international use
- One **Unicode** mapping uses 16 bits per character

2Bytes.

Code (Hex)	Character	Source
0041	A	English (Latin)
042F	Я	Russian (Cyrillic)
0E09	ฉ	Thai
13EA	Ꮺ	Cherokee
211E	℞	Letterlike symbols
21CC	⇌	Arrows
282F	⠯	Braille
345F	㑟	Chinese/Japanese/Korean (common)

FIGURE 3.6 A few characters in the Unicode character set

jeonghun.park@knu.ac.kr

Advanced Text Compression

- ASCII and unicode is a general, but not very efficient
  - One-to-one mapping between bits and each character

적은 Binary number로 mapping 하는게 효율 (등돌만 안쓰게)

- There are several advanced compression techniques
  - Keyword encoding
  - Run-length encoding
  - **Huffman encoding** « 매우중요

특정집단만 쓰도록하면 general 하진 못해도 efficient 함.

jeonghun.park@knu.ac.kr

Keyword Encoding

상당히 많이 쓴다고 생각되는 word를 Symbol화 하면
bit 아낌

- Replace frequently used patterns of text with a single special character, such as

Run-length encoding

RRRR ----RWWAR (100)
ASCII로 표현시
7bit × 100 써야하는데
반복되는게 많으므로
Run-length encoding 사용

(7▢W ▢A ▢R ▢)
ASCII 보다 꼭 효율적인건 아니고
(몇개 있는지 적어야하는 비트 있음)

WORD	SYMBOL
as	^
the	~
and	+
that	$
must	&
well	%
these	#

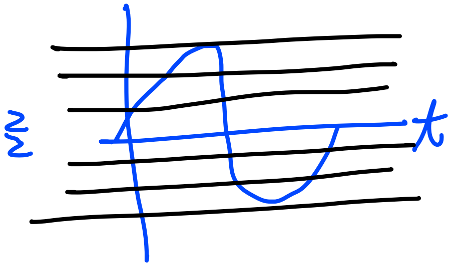
jeonghun.park@knu.ac.kr

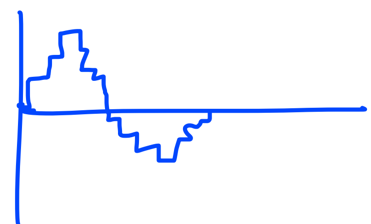
Audio Data Compression

- Sampling and Quantization
  - What is a difference?

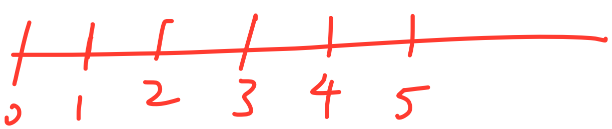
몇 초마다 이루어 지는지
(샘플링되는지)
결정

구간별로 나눠
속해있는 Amplitude를
표시





* 양자화만 해서
몇초 뒤 인지
알 수가 없음



세로로 끊기

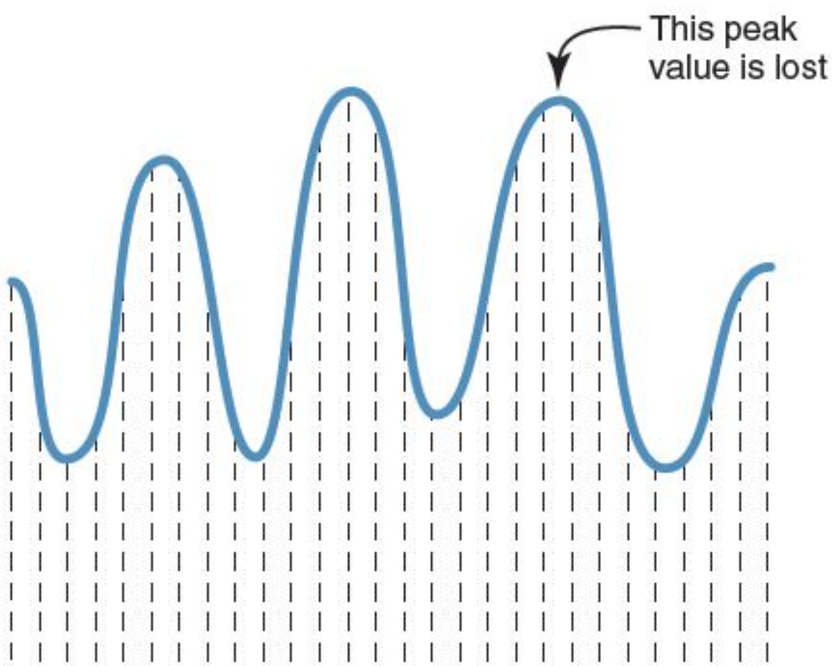


FIGURE 3.8 Sampling an audio signal

(maximum 주파수의 2배로 Sampling 하면
신호손실이 없음)

jeonghun.park@knu.ac.kr

How CDs Store Information

Binary를 물리적으로 새김!

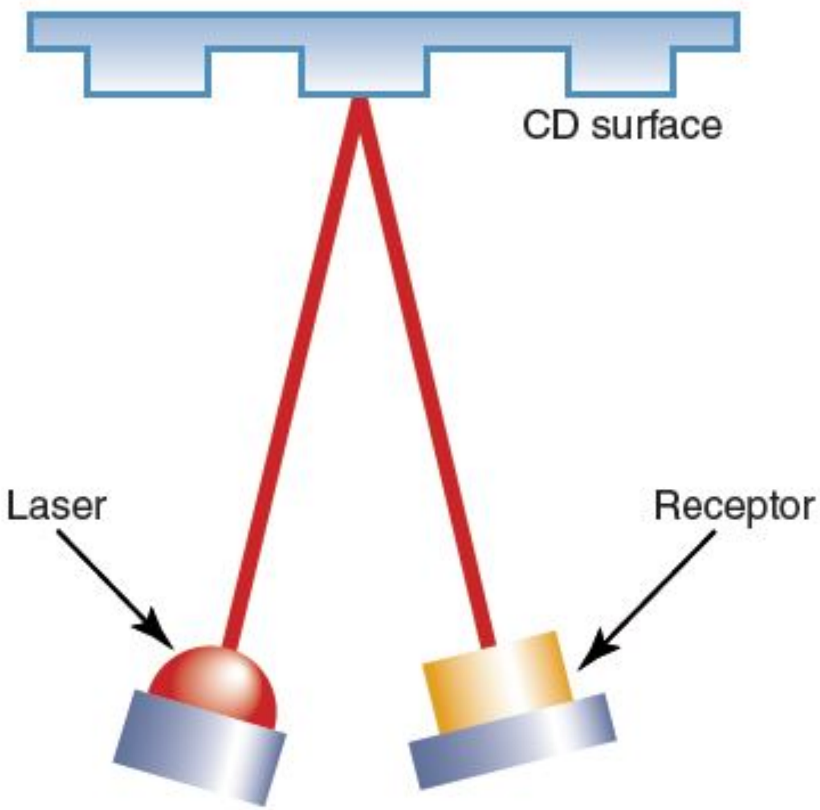


FIGURE 3.9 A CD player reading binary data

- CDs store audio (or other) information digitally

- Pits (reflect poorly) => 제대로 반사 안함(0)
- Lands (reflect well) => 반사 (1)

jeonghun.park@knu.ac.kr

Images and Colors Representations

저장된 데이터는 Binary 로 저장되었지만 우리한테 표현해야할때
어떻게 해야하는가?

- Retinas of our eyes have three types of **photoreceptor cone cells**
- Each type responds to a different set of frequencies of light
- Our brain translates that response into a perception of red, green, or blue

finite 하게 이미지 표현

jeonghun.park@knu.ac.kr
KNU KYUNGPOOK NATIONAL UNIVERSITY

Images and Colors (Contd.)

- Color is expressed as an RGB (red-green-blue) value – three numbers that indicate the relative contribution of each of these three primary colors
- For example, an RGB value of (255, 255, 0) maximizes the contribution of red and green, and minimizes the contribution of blue, which results in a bright yellow

jeonghun.park@knu.ac.kr

3-D Representation of Colors

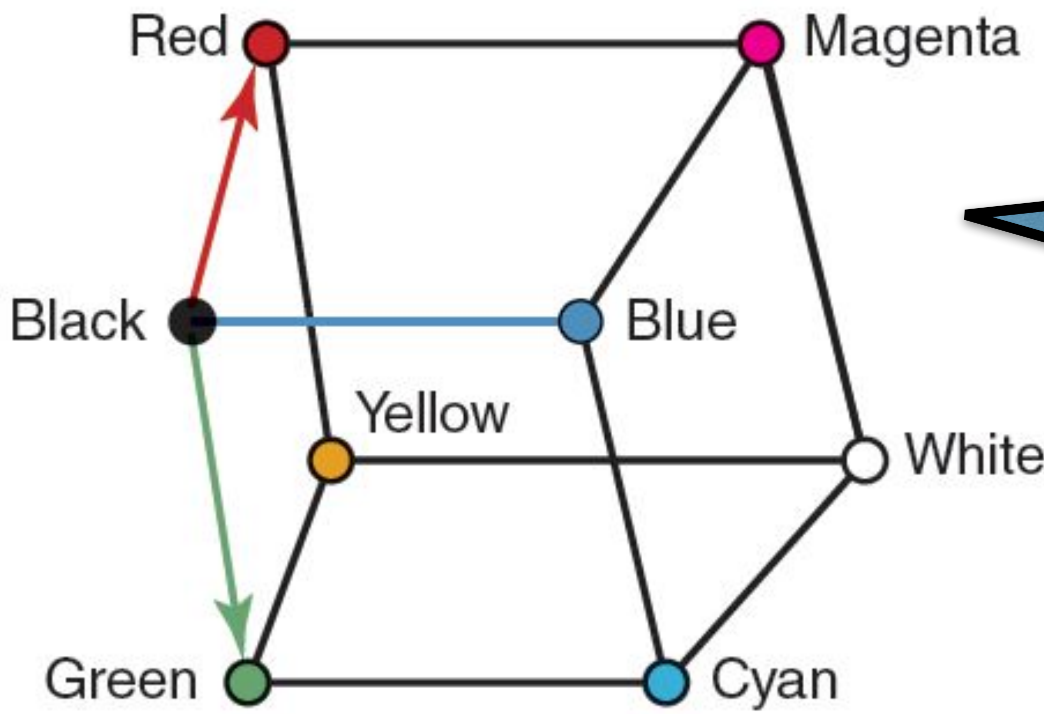
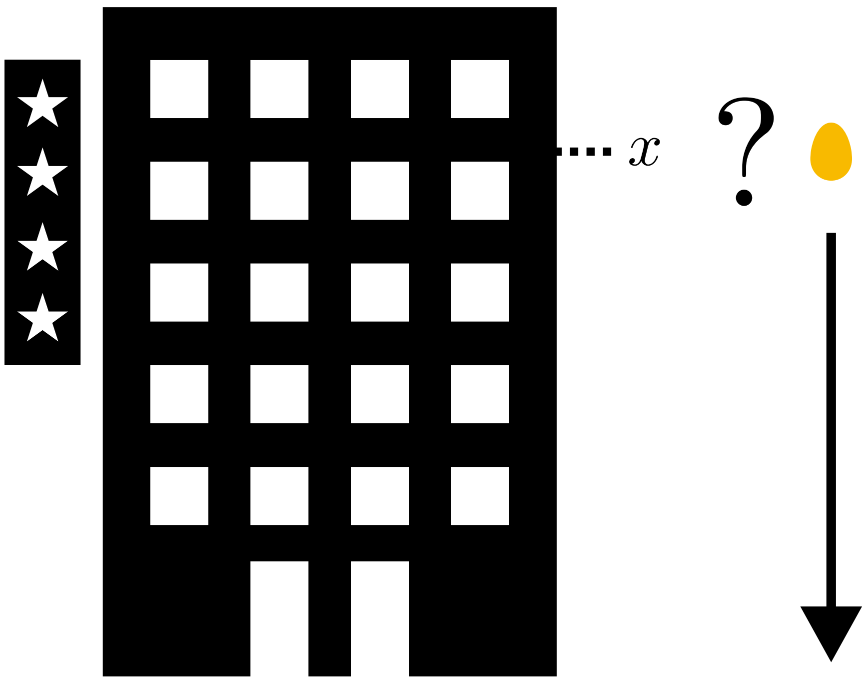


FIGURE 3.10 A three-dimensional color space

A combination of RGB values

jeonghun.park@knu.ac.kr

General Compression Theory



jeonghun.park@knu.ac.kr