Data Compression

Heiko Schwarz

Freie Universität Berlin
Fachbereich Mathematik und Informatik

Organization

Lecture: Monday 16:15-17:45
Room SR 006 / T9

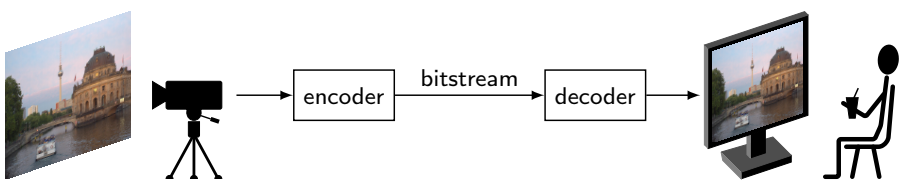
Exercise: Monday 14:30-16:00
Room SR 006 / T9

Web page: http://iphome.hhi.de/schwarz/DC.htm

Literature:

- Sayood, K. (2018), "**Introduction to Data Compression**," Morgan Kaufmann, Cambridge, MA.
- Cover, T. M. and Thomas, J. A. (2006), "**Elements of Information Theory**," John Wiley & Sons, New York.
- Gersho, A. and Gray, R. M. (1992), "**Vector Quantization and Signal Compression**," Kluwer Academic Publishers, Boston, Dordrecht, London.
- Jayant, N. S. and Noll, P. (1994), "**Digital Coding of Waveforms**," Prentice-Hall, Englewood Cliffs, NJ, USA.
- Wiegand, T. and Schwarz, H. (2010), "**Source Coding: Part I of Fundamentals of Source and Video Coding**," Foundations and Trends in Signal Processing, vol. 4, no. 1-2. (*pdf available on course web page*)

Introduction



Motivation for Data Compression (or Source Coding)

Motivation for Data Compression (or Source Coding)

Application of Data Compression

- Efficient transmission or storage of data

➡ **Use less throughput for given data**

➡ **Transmit more data for given throughput**

Motivation for Data Compression (or Source Coding)

Application of Data Compression

- Efficient transmission or storage of data

➞ **Use less throughput for given data**

➞ **Transmit more data for given throughput**

Data Compression: Enabling Technology

- Enables new applications
- Makes applications economically feasible

Motivation for Data Compression (or Source Coding)

Application of Data Compression

- Efficient transmission or storage of data

➞ **Use less throughput for given data**

➞ **Transmit more data for given throughput**

Data Compression: Enabling Technology

- Enables new applications
- Makes applications economically feasible

➞ **Well-known examples:**

- Distribution of digital images
- Distribution of digital audio
- Digitial television (DVB-T, DVB-T2, DVB-S, DVB-S2)
- Internet video streaming (YouTube, Netflix, Amazon, ...)

Practical Data Compression Examples

File Compression

- Compress large document archives

➜ WinZip, WinRar, gzip, lzip, bzip2, 7zip, ...

Practical Data Compression Examples

File Compression

- Compress large document archives

➜ WinZip, WinRar, gzip, lzip, bzip2, 7zip, ...

Audio Compression

- Compress music or audio books for storage on mobile devices

➜ FLAC, MP3, AAC, ...

Practical Data Compression Examples

File Compression

- Compress large document archives

➡ WinZip, WinRar, gzip, lzip, bzip2, 7zip, ...

Audio Compression

- Compress music or audio books for storage on mobile devices

➡ FLAC, MP3, AAC, ...

Image Compression

- Compress images for storage and distribution

➡ Raw camera formats, PNG, JPEG, JPEG-2000, JPEG-XR, H.265 | HEVC, ...

Practical Data Compression Examples

File Compression

- Compress large document archives

➜ WinZip, WinRar, gzip, lzip, bzip2, 7zip, ...

Audio Compression

- Compress music or audio books for storage on mobile devices

➜ FLAC, MP3, AAC, ...

Image Compression

- Compress images for storage and distribution

➜ Raw camera formats, PNG, JPEG, JPEG-2000, JPEG-XR, H.265 | HEVC, ...

Video Compression

- Compress video for streaming, broadcast, or storage

➜ MPEG-2, H.264 | AVC, VP9, H.265 | HEVC, ...

Types of Data Compression

Lossless Compression

- Invertible / reversible: Original input data can be completely recovered

Types of Data Compression

Lossless Compression

- Invertible / reversible: Original input data can be completely recovered

➡ Examples:
- gzip, lzip, 7szip, bzip2 for general files (most suitable for text)
- FLAC for audio
- PNG, JPEG-LS for images

Types of Data Compression

Lossless Compression

- Invertible / reversible: Original input data can be completely recovered

➡ Examples:
- gzip, lzip, 7szip, bzip2 for general files (most suitable for text)
- FLAC for audio
- PNG, JPEG-LS for images

Lossy Compression

- Not invertible: Only **approximation of original input data** can be recovered

Types of Data Compression

Lossless Compression

- Invertible / reversible: Original input data can be completely recovered

➡ Examples:
- gzip, lzip, 7szip, bzip2 for general files (most suitable for text)
- FLAC for audio
- PNG, JPEG-LS for images

Lossy Compression

- Not invertible: Only **approximation of original input data** can be recovered
- Achieves **much higher compression ratios**
- Dominant form of data compression for media

Types of Data Compression

Lossless Compression

- Invertible / reversible: Original input data can be completely recovered

➡ Examples:
- gzip, lzip, 7szip, bzip2 for general files (most suitable for text)
- FLAC for audio
- PNG, JPEG-LS for images

Lossy Compression

- Not invertible: Only **approximation of original input data** can be recovered
- Achieves **much higher compression ratios**
- Dominant form of data compression for media

➡ Examples:
- MP3, AAC for audio
- JPEG, JPEG-2000 for images
- MPEG-2, H.264 | AVC, H.265 | HEVC for video

Source Data

Analog Signals

- **Continuous-time/space** and **continuous-amplitude** signals
- Typically resulting from physical measurements
- Most signals in reality (e.g., audio and visual signals)

Source Data

Analog Signals

- **Continuous-time/space** and **continuous-amplitude** signals
- Typically resulting from physical measurements
- Most signals in reality (e.g., audio and visual signals)

Digital Signals

- **Discrete-time/space** and **discrete-amplitude** signals
- Often generated from analog signal
- Sometimes directly measured (e.g., image sensor)

Source Data

Analog Signals

- **Continuous-time/space** and **continuous-amplitude** signals
- Typically resulting from physical measurements
- Most signals in reality (e.g., audio and visual signals)

Digital Signals

- **Discrete-time/space** and **discrete-amplitude** signals
- Often generated from analog signal
- Sometimes directly measured (e.g., image sensor)

Input Data for Image and Video Coding

- Require **digital signals**
- Need to be stored and processed with a computer
- Compression methods are computer programs

Analog-to-Digital Conversion

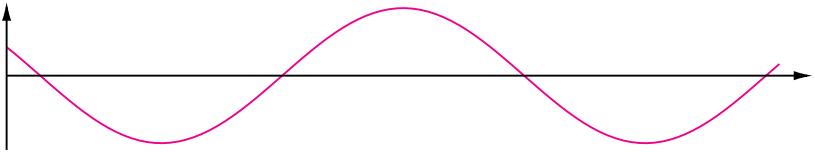
Sampling

- Maps continuous-time/space signal into discrete-time/space signal

Analog-to-Digital Conversion

Sampling

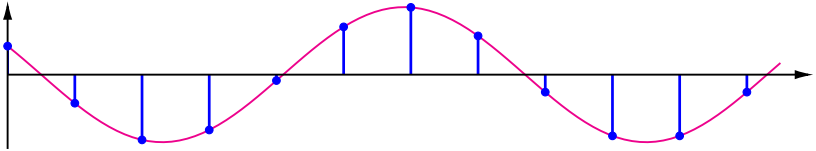
- Maps continuous-time/space signal into discrete-time/space signal



Analog-to-Digital Conversion

Sampling

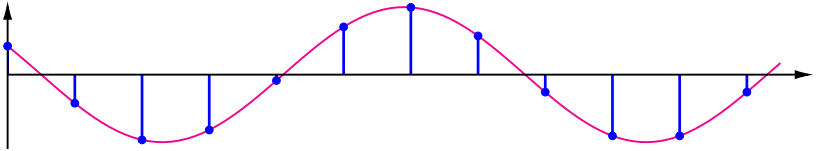
- Maps continuous-time/space signal into discrete-time/space signal



Analog-to-Digital Conversion

Sampling

- Maps continuous-time/space signal into discrete-time/space signal



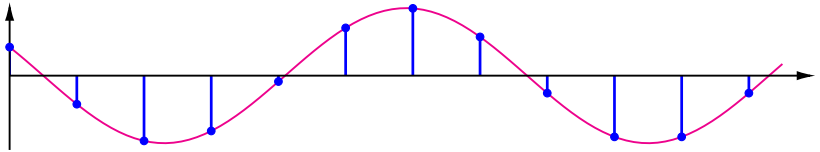
Quantization

- Maps continuous-amplitude signal into discrete-amplitude signal (e.g., by rounding)

Analog-to-Digital Conversion

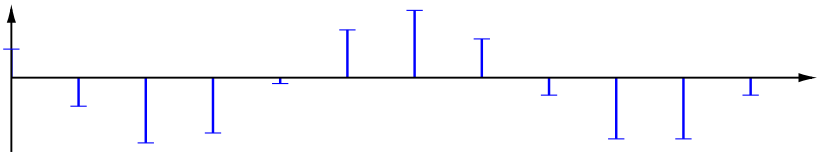
Sampling

- Maps continuous-time/space signal into discrete-time/space signal



Quantization

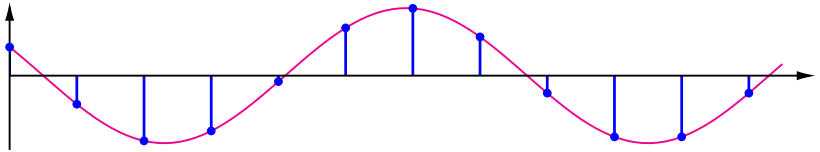
- Maps continuous-amplitude signal into discrete-amplitude signal (e.g., by rounding)



Analog-to-Digital Conversion

Sampling

- Maps continuous-time/space signal into discrete-time/space signal



Quantization

- Maps continuous-amplitude signal into discrete-amplitude signal (e.g., by rounding)

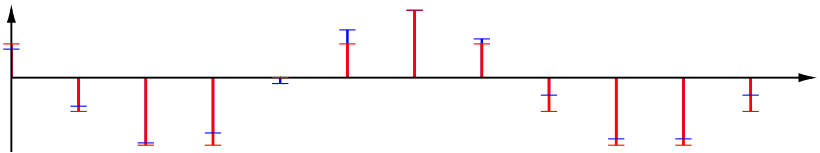
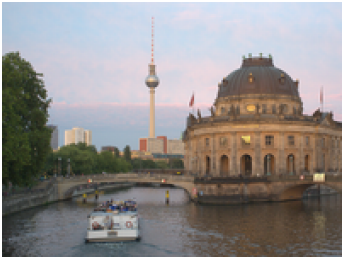


Image Example: Impact of Sampling (Spatial Resolution)

400×300 samples

Image Example: Impact of Sampling (Spatial Resolution)

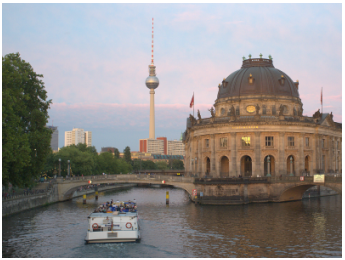
400×300 samples

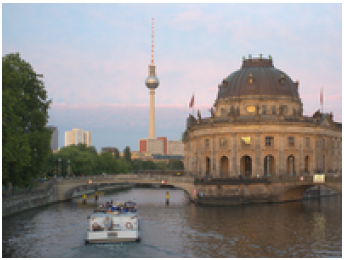


200×150 samples

Image Example: Impact of Sampling (Spatial Resolution)

400×300 samples





200×150 samples

100×75 samples

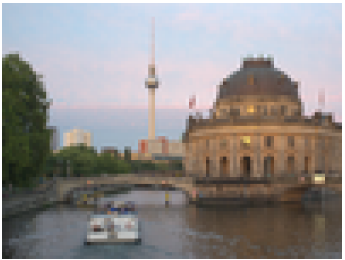
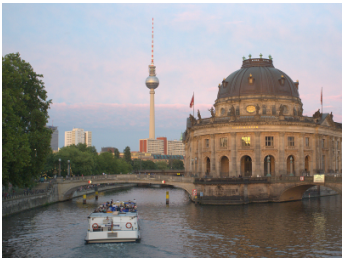
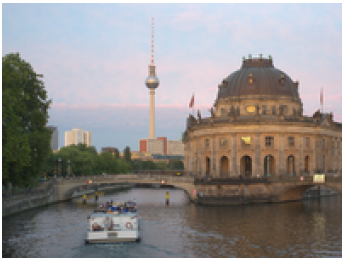


Image Example: Impact of Sampling (Spatial Resolution)

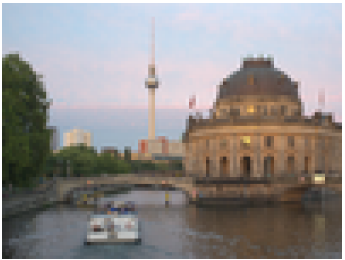
400×300 samples





200×150 samples

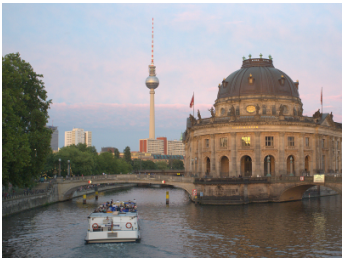
100×75 samples



50×38 samples

Image Example: Impact of Sampling (Spatial Resolution)

400×300 samples



200×150 samples

(interpolated to 400×300)

100×75 samples

(interpolated to 400×300)

50×38 samples

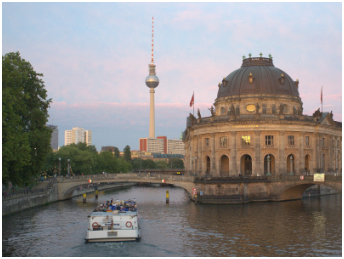
(interpolated to 400×300)

Image Example: Impact of Quantization (Bit Depth)

8 bits per component

Image Example: Impact of Quantization (Bit Depth)

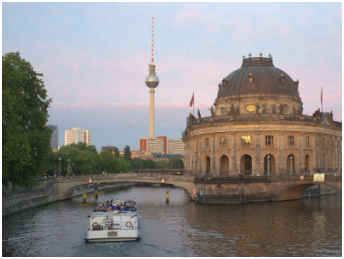
8 bits per component



6 bits per component

Image Example: Impact of Quantization (Bit Depth)

8 bits per component

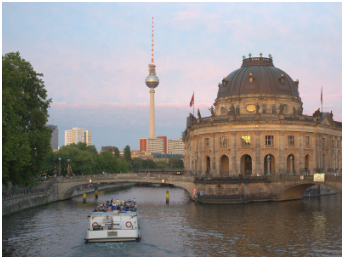


6 bits per component

4 bits per component

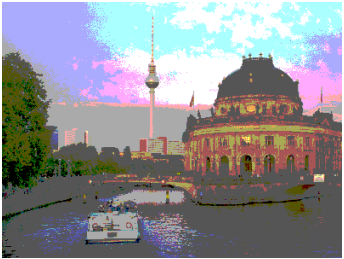
Image Example: Impact of Quantization (Bit Depth)

8 bits per component



6 bits per component

4 bits per component



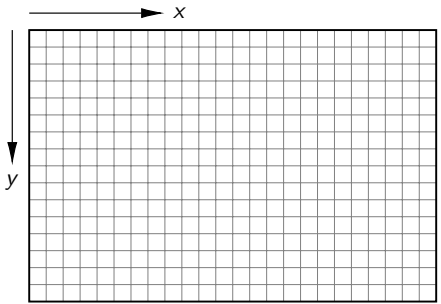
2 bits per component

Raw Images

Single-Component Image

- Matrix of integer samples

$$s[x, y]$$



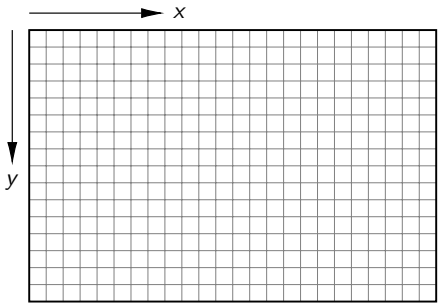
Raw Images

Single-Component Image

- Matrix of integer samples

$$s[x, y]$$

- Characterized by
  - ➡ Number of samples in horizontal direction W
  - ➡ Number of samples in vertical direction H
  - ➡ Sample bit depth B



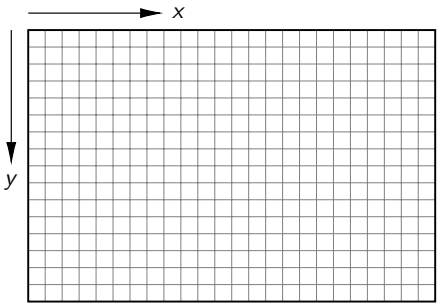
Raw Images

Single-Component Image

- Matrix of integer samples

$$s[x, y]$$

- Characterized by
  - ➡ Number of samples in horizontal direction W
  - ➡ Number of samples in vertical direction H
  - ➡ Sample bit depth B



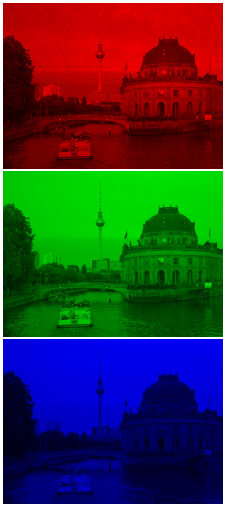
Color Images

- Three color components (typically RGB or YCbCr)

➡ Additionally characterized by color sampling format

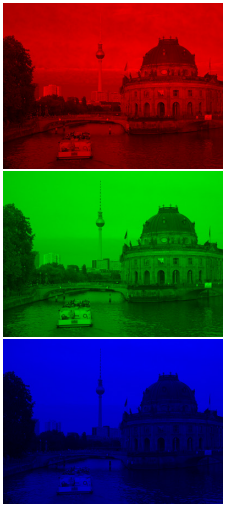
Color Sampling Formats

RGB



Color Sampling Formats

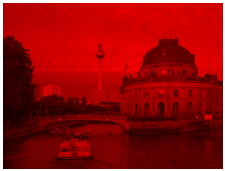
RGB

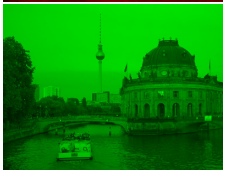


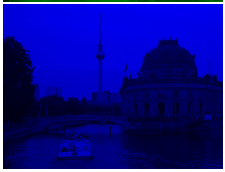
YCbCr 4:4:4

Color Sampling Formats

RGB





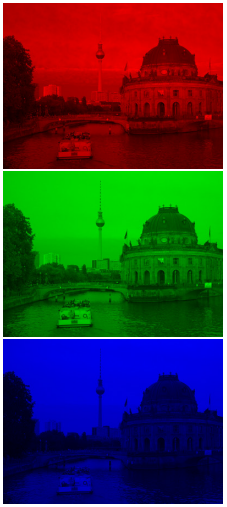


YCbCr 4:4:4

YCbCr 4:2:2

Color Sampling Formats

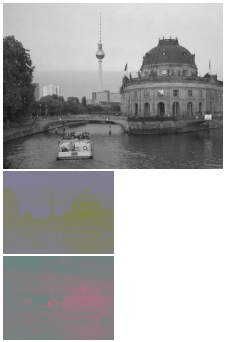
RGB



YCbCr 4:4:4

YCbCr 4:2:2

YCbCr 4:2:0



most common format

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels

➡ raw data rate $= 2$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second

➡ raw data rate $= 2 \cdot 44100\,\text{Hz}$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits}$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

- 50 pictures per second (Europe)

➡ raw data rate $= 50\,\text{Hz}$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

- 50 pictures per second (Europe)
- 3 color components

➡ raw data rate $= 50\,\text{Hz} \cdot 3$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

- 50 pictures per second (Europe)
- 3 color components with 3840×2160 samples each

➡ raw data rate $= 50\,\text{Hz} \cdot 3 \cdot 3840 \cdot 2160$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

- 50 pictures per second (Europe)
- 3 color components with 3840×2160 samples each
- 10 bits per sample

➡ raw data rate $= 50\,\text{Hz} \cdot 3 \cdot 3840 \cdot 2160 \cdot 10\,\text{bits}$

Raw Data Rate

Raw Data Rate: Bit rate R of raw data format

$$R = (\text{samples per time unit}) \cdot (\text{bit depth per sample})$$

Example: Stereo Audio Signal (CD Quality)

- 2 channels with 44 100 samples per second
- 16 bits per sample

➡ raw data rate $= 2 \cdot 44100\,\text{Hz} \cdot 16\,\text{bits} \approx \mathbf{1400\,kbits/s}$

Example: Ultra High Definition (UHD) Video

- 50 pictures per second (Europe)
- 3 color components with 3840×2160 samples each
- 10 bits per sample

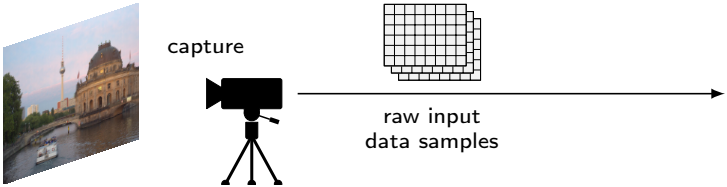
➡ raw data rate $= 50\,\text{Hz} \cdot 3 \cdot 3840 \cdot 2160 \cdot 10\,\text{bits} \approx \mathbf{12.4\,Gbits/s}$

Typical Communication Scenario

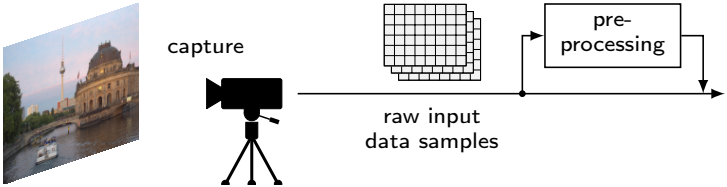
Typical Communication Scenario

capture

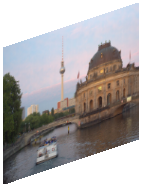
Typical Communication Scenario

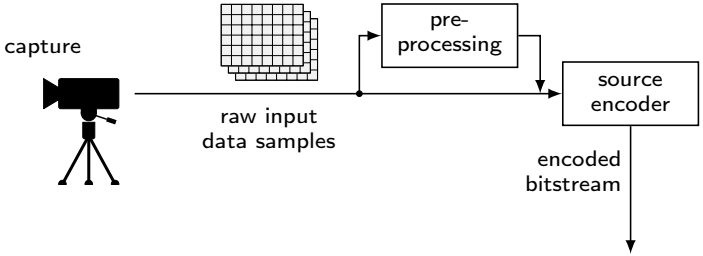


Typical Communication Scenario

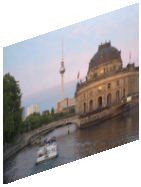


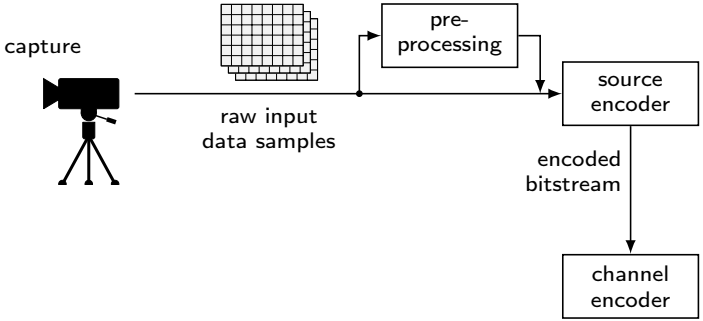
Typical Communication Scenario



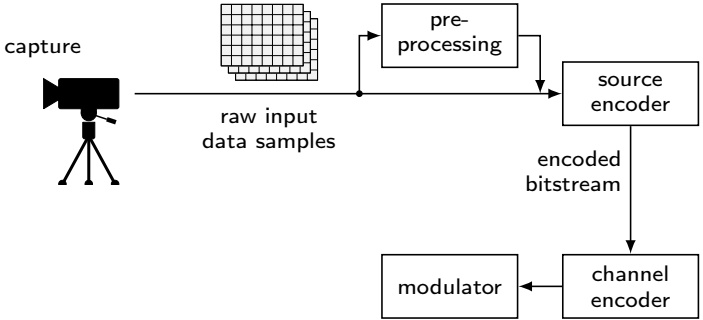


Typical Communication Scenario

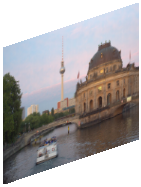


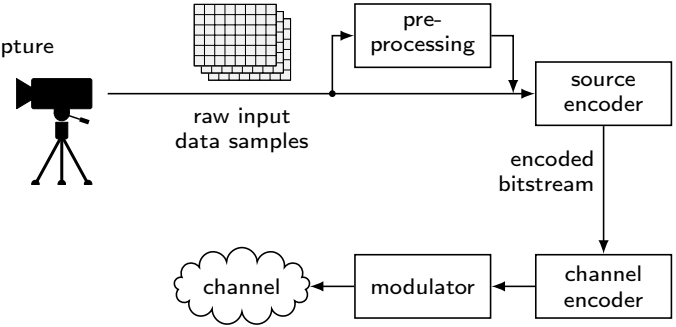


Typical Communication Scenario

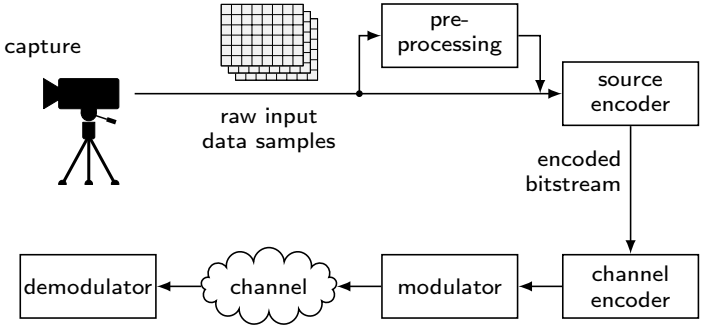


Typical Communication Scenario

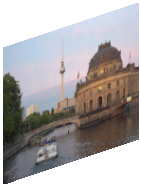


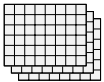


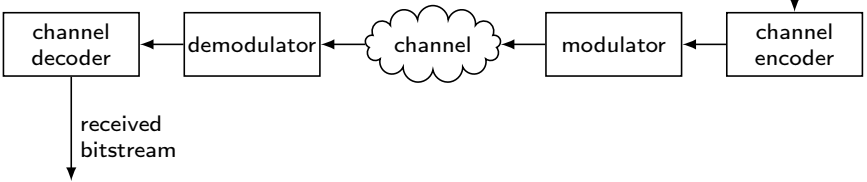
Typical Communication Scenario



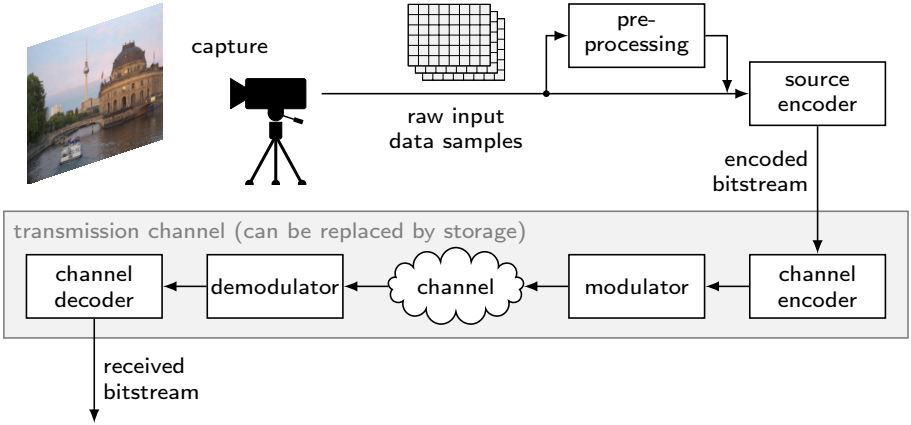
Typical Communication Scenario



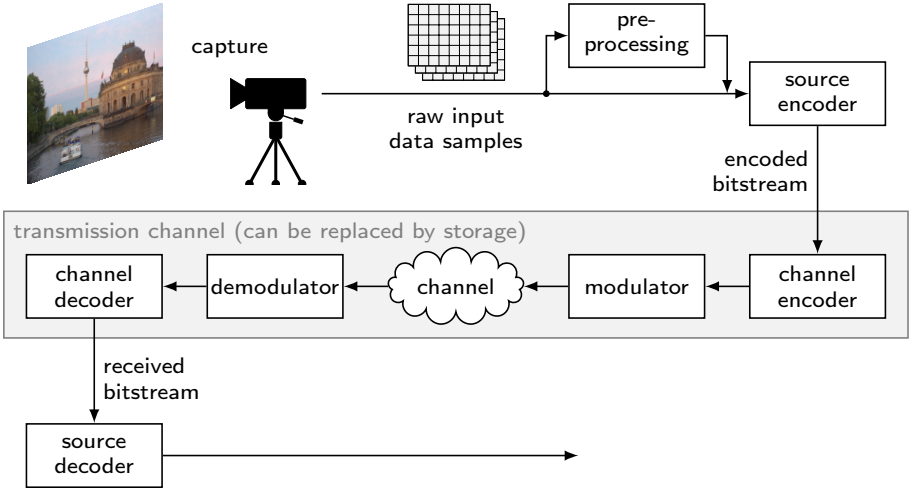




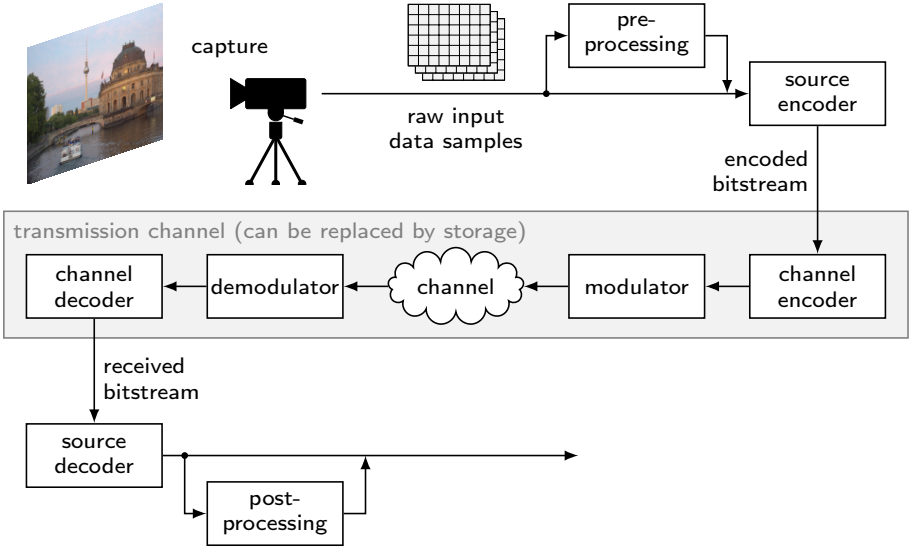
Typical Communication Scenario



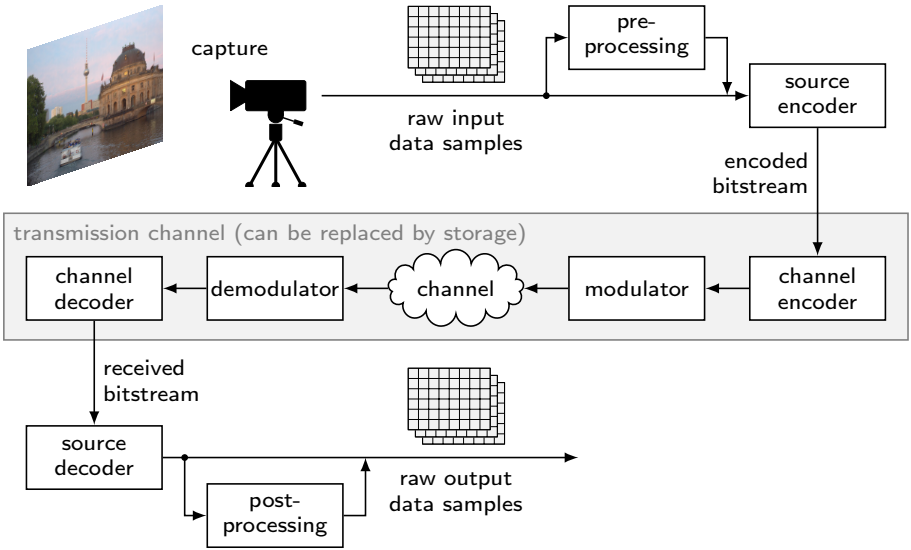
Typical Communication Scenario



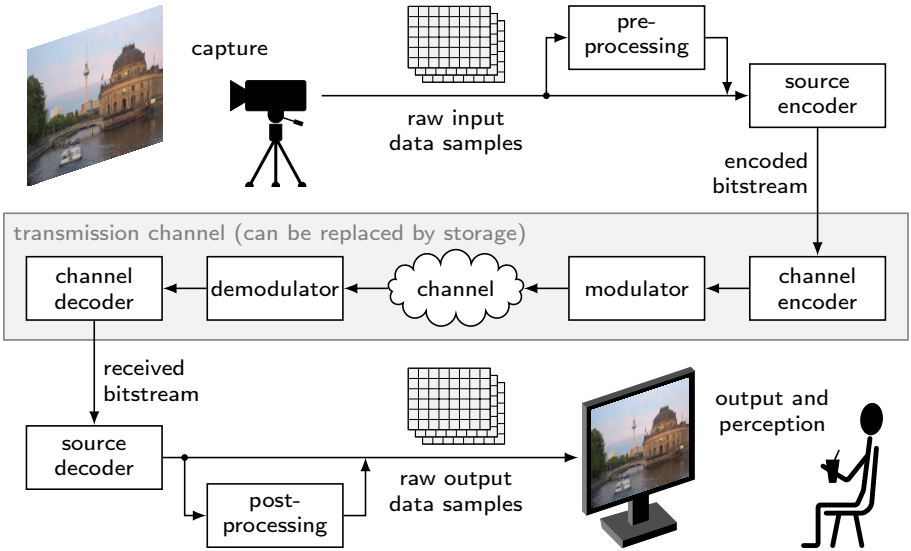
Typical Communication Scenario



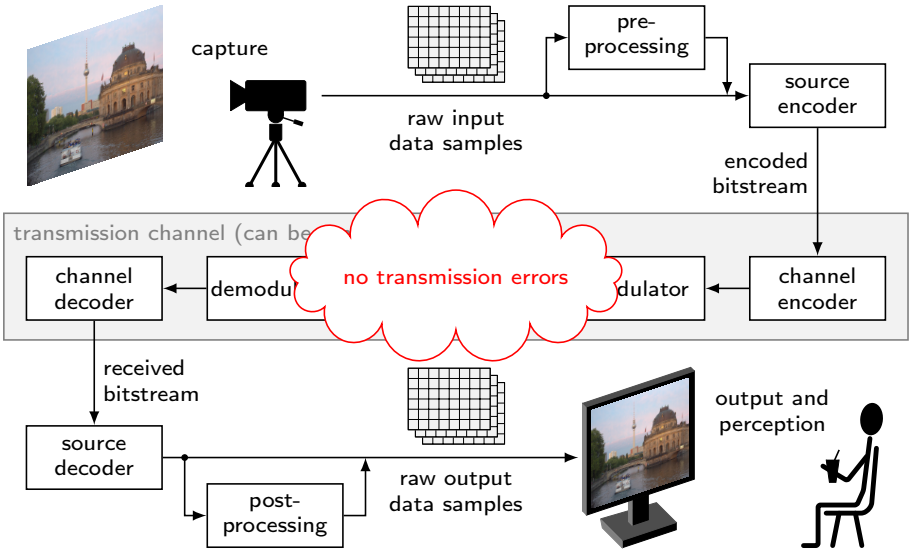
Typical Communication Scenario



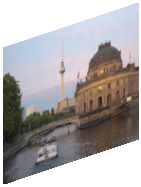
Typical Communication Scenario



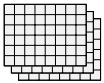
Typical Communication Scenario



Typical Communication Scenario



capture



raw input
data samples

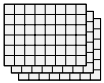
pre-
processing

source
encoder

bitstream

source
decoder

post-
processing



raw output
data samples

output and
perception

The Basic Source Coding Problem

Basic Source Coding Problem

- Two equivalent formulations:

Representing source data with highest fidelity possible within an available bit rate

The Basic Source Coding Problem

Basic Source Coding Problem

- Two equivalent formulations:

Representing source data with highest fidelity possible within an available bit rate

and

Representing source data using lowest bit rate possible while maintaining a specified reproduction quality

The Basic Source Coding Problem

Basic Source Coding Problem

- Two equivalent formulations:

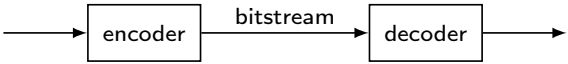
Representing source data with highest fidelity possible within an available bit rate

and

Representing source data using lowest bit rate possible while maintaining a specified reproduction quality

Source Codec

- Source **codec**: System of en**co**der and **dec**oder



Practical Source Coding Problem

Characteristics of Source Codecs

- Bit rate: Throughput of the communication channel
- Quality: Fidelity of the reconstructed signal
- Delay: Start-up latency, end-to-end delay
- Complexity: Computation, memory, memory access

Practical Source Coding Problem

Characteristics of Source Codecs

- Bit rate: Throughput of the communication channel
- Quality: Fidelity of the reconstructed signal
- Delay: Start-up latency, end-to-end delay
- Complexity: Computation, memory, memory access

Practical Source Coding Problem

Given a maximum allowed complexity and a maximum delay, achieve an optimal trade-off between bit rate and reconstruction quality for the transmission problem in the targeted application

Practical Source Coding Problem

Characteristics of Source Codecs

- Bit rate: Throughput of the communication channel
- Quality: Fidelity of the reconstructed signal
- Delay: Start-up latency, end-to-end delay
- Complexity: Computation, memory, memory access

Practical Source Coding Problem

Given a maximum allowed complexity and a maximum delay, achieve an optimal trade-off between bit rate and reconstruction quality for the transmission problem in the targeted application

In this course:

➡ Will concentrate on source codec

➡ Ignore aspects of transmission channel (e.g., transmission errors)

Text Compression

		Goethe.txt (19 books)		**VTM-6.0-source.tar**	
utility	**basic compression algorithm**	file size (bytes)	factor	file size (bytes)	factor
uncompressed original (1 byte per character)		4 932 772	1.00	6 522 880	1.00

Text Compression

utility	basic compression algorithm	**Goethe.txt** (19 books) file size (bytes)	factor	**VTM-6.0-source.tar** file size (bytes)	factor
uncompressed original (1 byte per character)		4 932 772	1.00	6 522 880	1.00
pack	Huffman	2 560 960	1.93	2 054 989	3.17
compress	Lempel-Ziv (LZW)	2 028 603	2.43	2 364 415	2.76
zip	Lempel-Ziv (LZSS) + Huffman	1 895 119	2.60	1 253 504	5.20
gzip	Lempel-Ziv (LZSS) + Huffman	1 894 978	2.60	1 253 356	5.20
rar	Lempel-Ziv (LZSS) + Huffman	1 518 071	3.25	950 039	6.87
xz	Lempel-Ziv (LZMA)	1 397 676	3.53	858 196	7.60
7zip	Lempel-Ziv (LZMA)	1 397 686	3.53	879 900	7.41
lzip	Lempel-Ziv (LZMA)	1 396 587	3.53	850 044	7.67
bzip2	Burrows-Wheeler + Huffman	1 389 030	3.55	1 035 500	6.30

Text Compression

utility	basic compression algorithm	**Goethe.txt** (19 books) file size (bytes)	factor	**VTM-6.0-source.tar** file size (bytes)	factor
uncompressed original (1 byte per character)		4 932 772	1.00	6 522 880	1.00
pack	Huffman	2 560 960	1.93	2 054 989	3.17
compress	Lempel-Ziv (LZW)	2 028 603	2.43	2 364 415	2.76
zip	Lempel-Ziv (LZSS) + Huffman	1 895 119	2.60	1 253 504	5.20
gzip	Lempel-Ziv (LZSS) + Huffman	1 894 978	2.60	1 253 356	5.20
rar	Lempel-Ziv (LZSS) + Huffman	1 518 071	3.25	950 039	6.87
xz	Lempel-Ziv (LZMA)	1 397 676	3.53	858 196	7.60
7zip	Lempel-Ziv (LZMA)	1 397 686	3.53	879 900	7.41
lzip	Lempel-Ziv (LZMA)	1 396 587	3.53	850 044	7.67
bzip2	Burrows-Wheeler + Huffman	1 389 030	3.55	1 035 500	6.30
cmix (v18)	arithmetic coding with model prediction (extremely complex: multiple hours for en-/decoding)	933 717	5.28	452 539	14.41

Audio Compression

compression	file size (bytes)	factor	file size (bytes)	factor
original (CD)	501 763 964	1.00	628 191 020	1.00

Audio Compression

compression	file size (bytes)	factor	file size (bytes)	factor
original (CD)	501 763 964	1.00	628 191 020	1.00
lzip	428 713 583	1.17	466 668 657	1.35
bzip2	437 391 898	1.15	466 695 240	1.35

Audio Compression

compression	file size (bytes)	factor	file size (bytes)	factor
original (CD)	501 763 964	1.00	628 191 020	1.00
lzip	428 713 583	1.17	466 668 657	1.35
bzip2	437 391 898	1.15	466 695 240	1.35
FLAC (lossless)	276 312 806	1.82	335 378 545	1.87

Audio Compression

compression	file size (bytes)	factor	file size (bytes)	factor	
original (CD)	501 763 964	1.00	628 191 020	1.00	
lzip	428 713 583	1.17	466 668 657	1.35	
bzip2	437 391 898	1.15	466 695 240	1.35	
FLAC (lossless)	276 312 806	1.82	335 378 545	1.87	
MP3 (256 kbit/s: excellent)	91 024 822	5.51	113 959 914	5.51	play
MP3 (64 kbit/s: acceptable)	22 756 416	22.05	28 490 112	22.05	play
MP3 (16 kbit/s: poor)	5 689 296	88.19	7 122 672	88.19	play

Image Compression

RGB: 960×720

RGB: 1024×640

compression	file size (bytes)	factor	file size (bytes)	factor
original (8 bit per sample)	2 073 600	1.00	1 966 080	1.00

Image Compression

RGB: 960×720

RGB: 1024×640

compression	file size (bytes)	factor	file size (bytes)	factor
original (8 bit per sample)	2 073 600	1.00	1 966 080	1.00
lzip	1 469 323	1.41	1 084 025	1.81
bzip2	1 489 411	1.39	1 050 415	1.87

Image Compression

RGB: 960×720

RGB: 1024×640

compression	file size (bytes)	factor	file size (bytes)	factor
original (8 bit per sample)	2 073 600	1.00	1 966 080	1.00
lzip	1 469 323	1.41	1 084 025	1.81
bzip2	1 489 411	1.39	1 050 415	1.87
PNG (lossless image coding)	1 280 650	1.62	907 330	2.17

Image Compression

RGB: 960×720

RGB: 1024×640

compression	file size (bytes)	factor	file size (bytes)	factor
original (8 bit per sample)	2 073 600	1.00	1 966 080	1.00
lzip	1 469 323	1.41	1 084 025	1.81
bzip2	1 489 411	1.39	1 050 415	1.87
PNG (lossless image coding)	1 280 650	1.62	907 330	2.17
JPEG (lossy: very good)	61 197	33.88	40 012	49.14
JPEG (lossy: acceptable)	40 609	51.06	31 058	63.30

Image Compression: Quality versus Compression Ratio

Original Image (1024 × 640 image points, 1966 KB)

100 %

Image Compression: Quality versus Compression Ratio

Lossless Compressed: LZIP

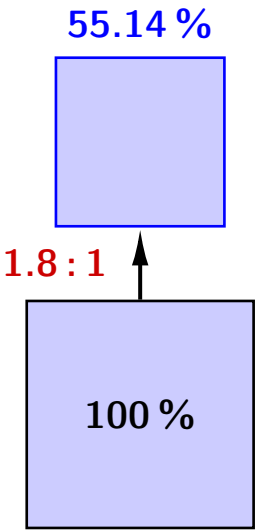


Image Compression: Quality versus Compression Ratio

Lossless Compressed: PNG

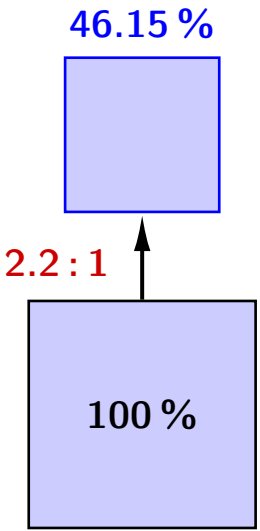


Image Compression: Quality versus Compression Ratio

Lossy Compressed: JPEG (Quality 95)

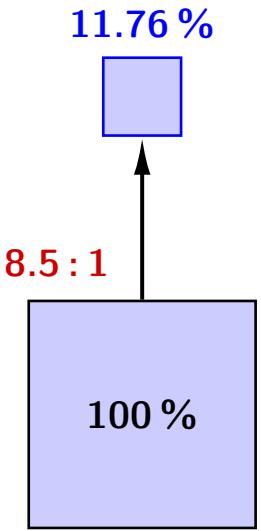


Image Compression: Quality versus Compression Ratio

Lossy Compressed: JPEG (Quality 75)

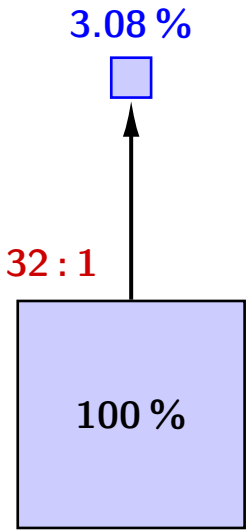


Image Compression: Quality versus Compression Ratio

Lossy Compressed: JPEG (Quality 50)

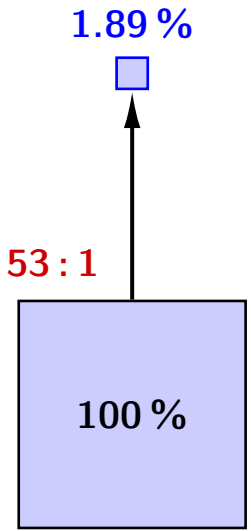


Image Compression: Quality versus Compression Ratio

Lossy Compressed: JPEG (Quality 25)

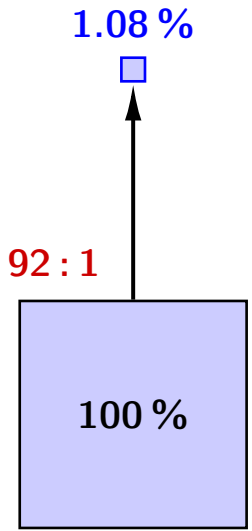
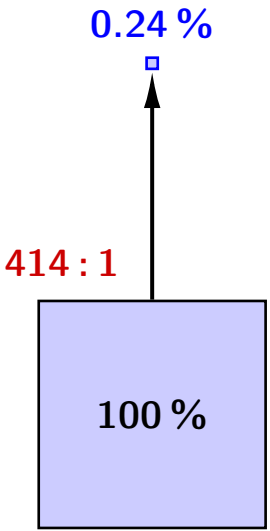


Image Compression: Quality versus Compression Ratio

Lossy Compressed: JPEG (Quality 1)



Measuring Coding Efficiency

Average Bit Rate

$$\text{images:} \qquad R = \frac{\text{number of bits in bitstream for the image}}{\text{number of pixels in the image}}$$

Measuring Coding Efficiency

Average Bit Rate

$$\text{images:} \qquad R = \frac{\text{number of bits in bitstream for the image}}{\text{number of pixels in the image}}$$

$$\text{audio/video:} \qquad R = \frac{\text{number of bits in bitstream for the audio/video sequence}}{\text{nominal duration of the audio/video sequence}}$$

Measuring Coding Efficiency

Average Bit Rate

$$\text{images:} \qquad R = \frac{\text{number of bits in bitstream for the image}}{\text{number of pixels in the image}}$$

$$\text{audio/video:} \qquad R = \frac{\text{number of bits in bitstream for the audio/video sequence}}{\text{nominal duration of the audio/video sequence}}$$

Quality / Distortion

- Ideally: Quality as perceived by human beings

Measuring Coding Efficiency

Average Bit Rate

$$\text{images:} \qquad R = \frac{\text{number of bits in bitstream for the image}}{\text{number of pixels in the image}}$$

$$\text{audio/video:} \qquad R = \frac{\text{number of bits in bitstream for the audio/video sequence}}{\text{nominal duration of the audio/video sequence}}$$

Quality / Distortion

- Ideally: Quality as perceived by human beings
- Often used: Mean square error (MSE) and peak-signal-to-noise ratio (PSNR)

$$\mathrm{MSE} = \frac{1}{N} \sum_{\forall k} \Big(s[k] - s'[k] \Big)^2$$

$s[k]$ – original samples
$s'[k]$ – reconstructed samples
N – number of samples

Measuring Coding Efficiency

Average Bit Rate

$$\text{images:} \qquad R = \frac{\text{number of bits in bitstream for the image}}{\text{number of pixels in the image}}$$

$$\text{audio/video:} \qquad R = \frac{\text{number of bits in bitstream for the audio/video sequence}}{\text{nominal duration of the audio/video sequence}}$$

Quality / Distortion

- Ideally: Quality as perceived by human beings
- Often used: Mean square error (MSE) and peak-signal-to-noise ratio (PSNR)

$$\text{MSE} = \frac{1}{N} \sum_{\forall k} \Big(s[k] - s'[k] \Big)^2$$

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{(2^B - 1)^2}{\text{MSE}} \right)$$

$s[k]$ – original samples
$s'[k]$ – reconstructed samples
N – number of samples
B – sample bit depth

Trade-Off between Quality and Compression Ratio

Coding Efficiency

- Ability to trade-off bit rate and reconstruction quality

Trade-Off between Quality and Compression Ratio

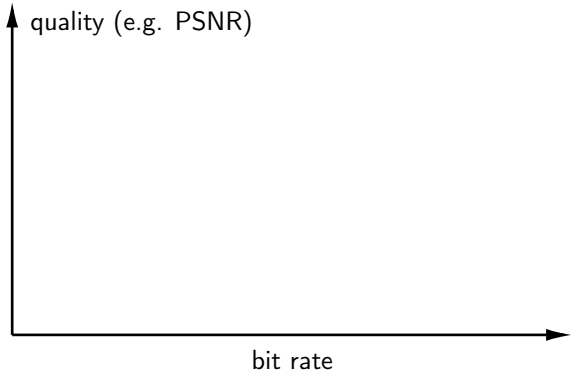
Coding Efficiency

- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)

Trade-Off between Quality and Compression Ratio

Coding Efficiency

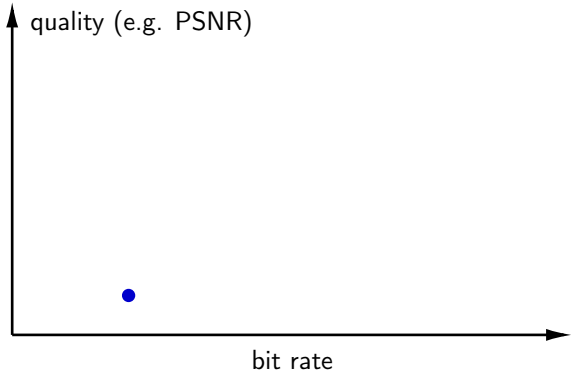
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

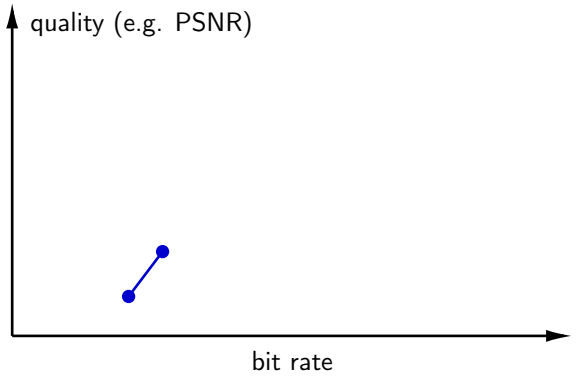
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

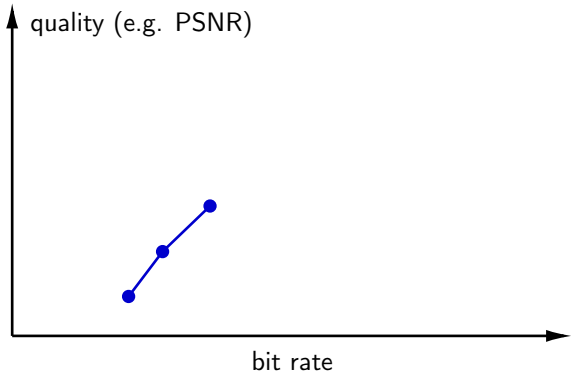
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

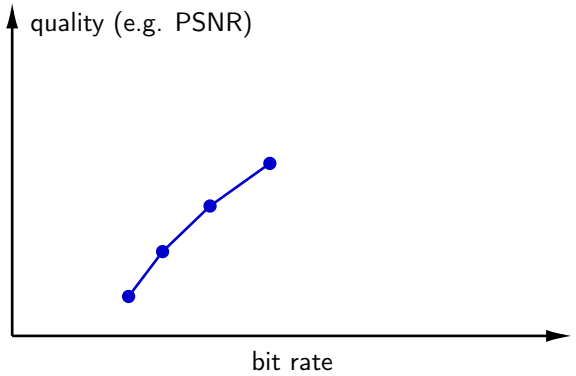
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

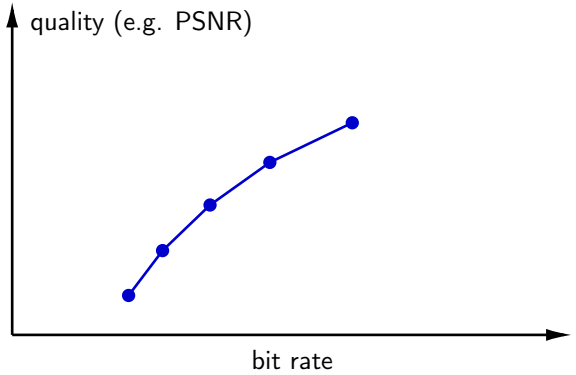
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

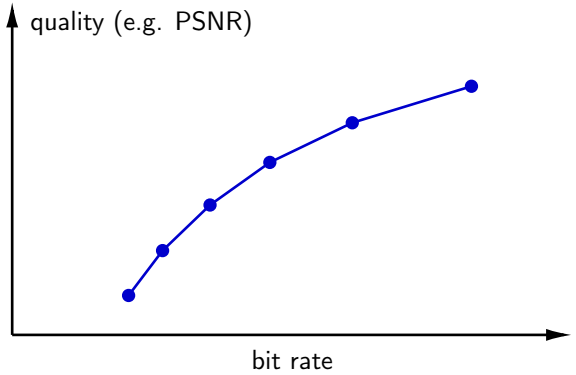
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

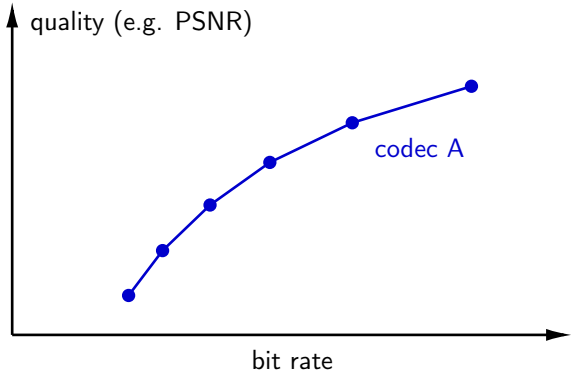
- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)



Trade-Off between Quality and Compression Ratio

Coding Efficiency

- Ability to trade-off bit rate and reconstruction quality
- Want **best reconstruction quality for a given bitrate** (or vice versa)

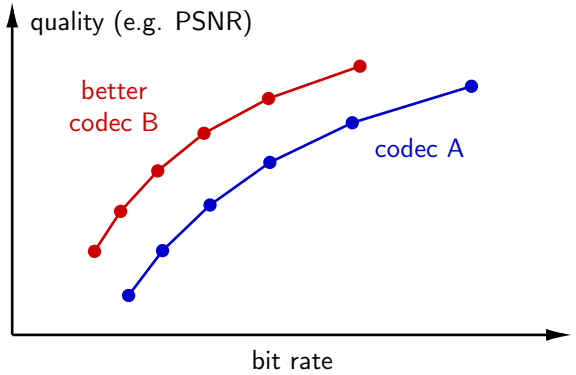


Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 5:1 compression

JPEG (42.5 dB @ 5:1)

VVC (54.2 dB @ 5:1)

Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 12:1 compression

JPEG (39.3 dB @ 12:1)

VVC (45.6 dB @ 12:1)

Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 25:1 compression

JPEG (34.6 dB @ 26:1)

VVC (39.9 dB @ 26:1)

Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 50:1 compression

JPEG (31.4 dB @ 54:1)

VVC (35.7 dB @ 55:1)

Image Compression: JPEG versus VVC (VTM-6.0)

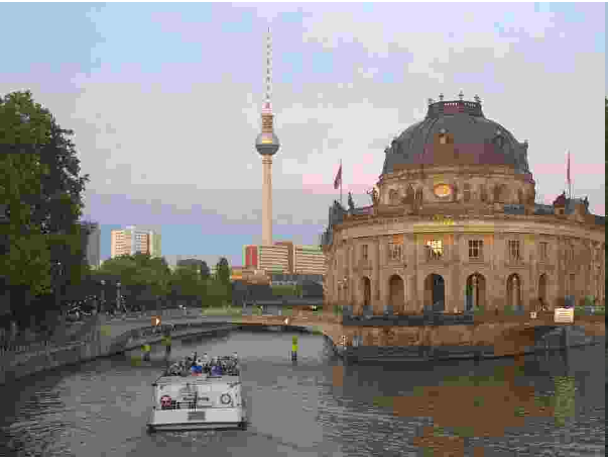
$\approx$ 100:1 compression

JPEG (29.1 dB @ 107:1)

VVC (32.6 dB @ 114:1)

Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 200:1 compression



JPEG (26.9 dB @ 204:1)

VVC (30.4 dB @ 223:1)

Image Compression: JPEG versus VVC (VTM-6.0)

$\approx$ 350:1 compression

JPEG (24.1 dB @ 357:1)

VVC (28.9 dB @ 381:1)

Video Compression

	HD movie on Blu-ray disc	UHD broadcast over DVB-S2
raw video format	1920×1024 image points 24 pictures per second 3×8 bits per image point	3840×2160 image points 50 pictures per second 3×10 bits per image point
raw data rate	**ca. 1140 Mbits/s**	**ca. 12 GBits/s**

Video Compression

	HD movie on Blu-ray disc	**UHD broadcast over DVB-S2**
raw video format	1920×1024 image points 24 pictures per second 3×8 bits per image point	3840×2160 image points 50 pictures per second 3×10 bits per image point
raw data rate	**ca. 1140 Mbits/s**	**ca. 12 GBits/s**
channel bit rate	36 Mbits/s (read speed)	58 MBit/s (8PK5 2/3)
video bit rate	**ca. 20 MBits/s**	**ca. 25 Mbits/s**

Video Compression

	HD movie on Blu-ray disc	UHD broadcast over DVB-S2
raw video format	1920×1024 image points 24 pictures per second 3×8 bits per image point	3840×2160 image points 50 pictures per second 3×10 bits per image point
raw data rate	**ca. 1140 Mbits/s**	**ca. 12 GBits/s**
channel bit rate	36 Mbits/s (read speed)	58 MBit/s (8PK5 2/3)
video bit rate	**ca. 20 MBits/s**	**ca. 25 Mbits/s**
required compression	**ca. 60 : 1**	**ca. 500 : 1**

Video Compression: Quality versus Compression Ratio

H.264 | AVC @ 4493 kbit/s (107 : 1)

H.264 | AVC @ 414 kbit/s (1160 : 1)

Original: 479 232 kbit/s (832×480, 50Hz, RGB)

Video Compression: Quality versus Compression Ratio

H.264 | AVC @ 2702 kbit/s (178 : 1)

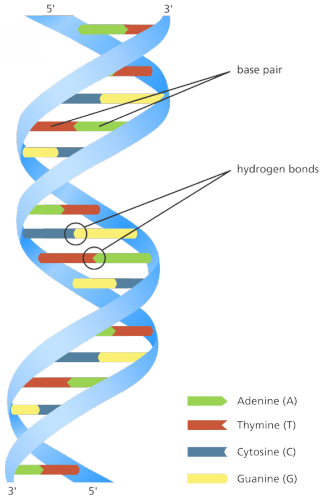
H.264 | AVC @ 415 kbit/s (1156 : 1)

Original: 479 232 kbit/s (832×480, 50Hz, RGB)

Genome Compression

Information in Human Genome

- Roughly 3 million base pairs in human genomes
- Four different base pairs (2 bits per base pair)
- ➡ Information in human genome $\approx$ **700 MBytes**



Genome Compression

Information in Human Genome

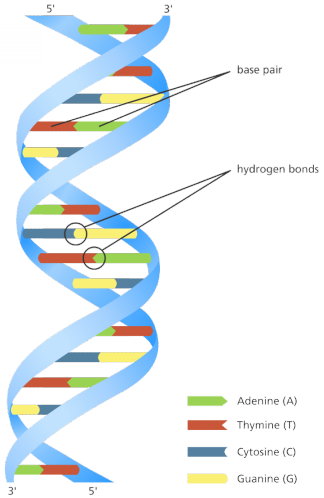
- Roughly 3 million base pairs in human genomes
- Four different base pairs (2 bits per base pair)

➡ Information in human genome $\approx$ **700 MBytes**

Data acquired by Genome Sequencer

- Sequence genome by a bunch of short reads

➡ Generated genome data $\approx$ **200 GBytes**



Genome Compression

Information in Human Genome

- Roughly 3 million base pairs in human genomes
- Four different base pairs (2 bits per base pair)

➡ Information in human genome $\approx$ **700 MBytes**

Data acquired by Genome Sequencer

- Sequence genome by a bunch of short reads

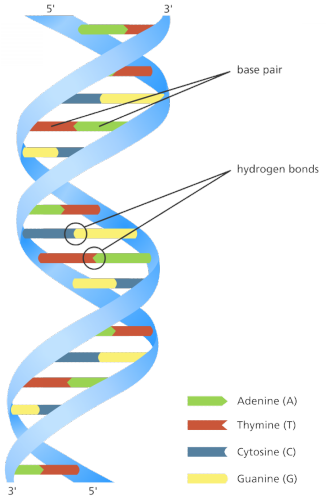
➡ Generated genome data $\approx$ **200 GBytes**

Compression of Genome Data

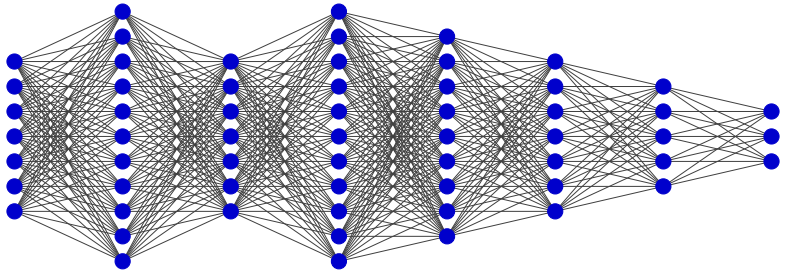
- Only about 0.1% of the genome differs among individuals

➡ Only need to code differences (i.e., "mutations")

➡ Compression format should allow simple comparisons



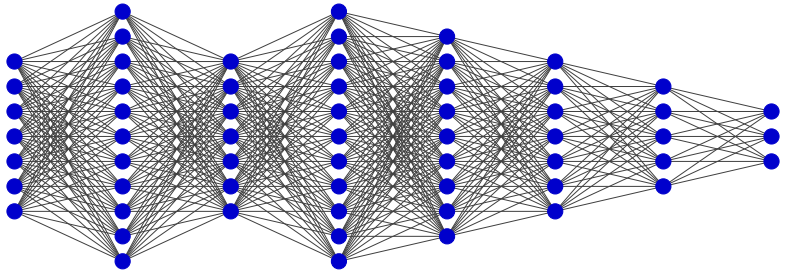
Compression of Neural Networks



Deep Neural Networks

- Example (VGG-16): 16 weight layers, 138 million parameters, 553 MB uncompressed
- Inference has high power consumption

Compression of Neural Networks



Deep Neural Networks

- Example (VGG-16): 16 weight layers, 138 million parameters, 553 MB uncompressed
- Inference has high power consumption
- Goal of compression:
  - ➡ Smaller representation with minimum impact on accuracy
  - ➡ Computational more efficient inference

JPEG Principle — Transform Coding of Sample Blocks

input video picture

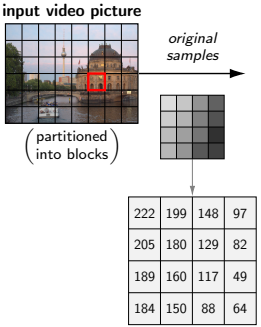
JPEG Principle — Transform Coding of Sample Blocks

input video picture

(partitioned into blocks)

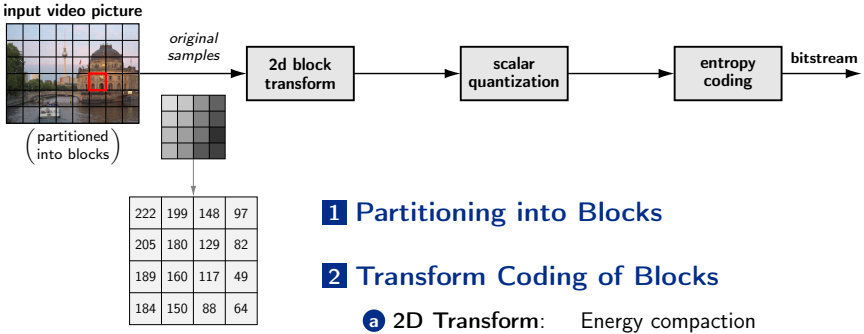
1 Partitioning into Blocks

JPEG Principle — Transform Coding of Sample Blocks



1 Partitioning into Blocks

JPEG Principle — Transform Coding of Sample Blocks

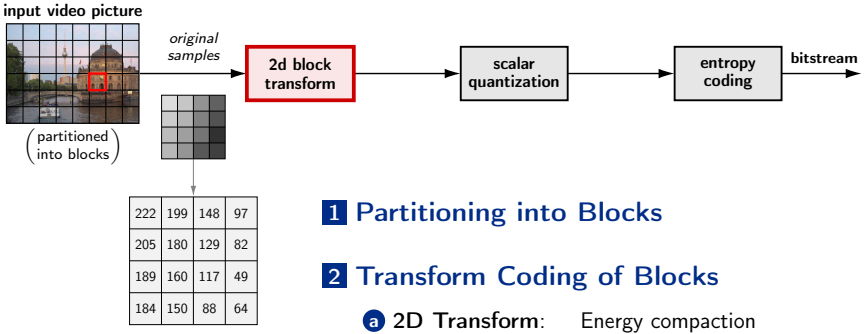


1 Partitioning into Blocks

2 Transform Coding of Blocks

- **a 2D Transform**: Energy compaction
- **b Quantization**: Approximate signal (remove invisible details)
- **c Entropy Coding**: Represent data with as little bits as possible

JPEG Principle — Transform Coding of Sample Blocks



1 Partitioning into Blocks

2 Transform Coding of Blocks

- **a 2D Transform**: Energy compaction
- **b Quantization**: Approximate signal (remove invisible details)
- **c Entropy Coding**: Represent data with as little bits as possible

Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)

Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)

Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)

Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

Transforms in Data Compression

- Most often: **Discrete Cosine Transform (DCT)** or approximation thereof

Orthogonal Block Transform

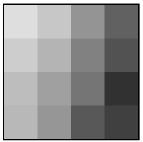
Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

Transforms in Data Compression

- Most often: **Discrete Cosine Transform (DCT)** or approximation thereof

original block



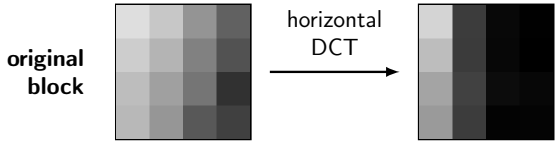
Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

Transforms in Data Compression

- Most often: **Discrete Cosine Transform (DCT)** or approximation thereof



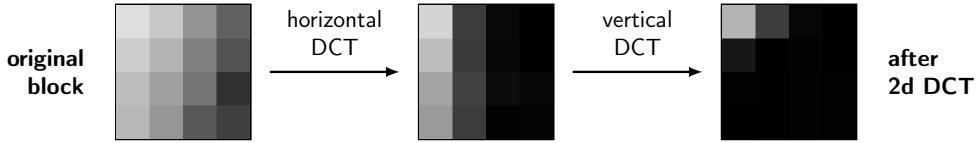
Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

Transforms in Data Compression

- Most often: **Discrete Cosine Transform (DCT)** or approximation thereof



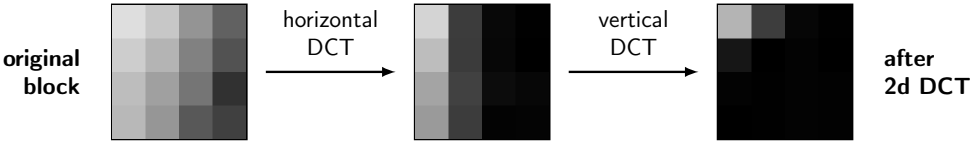
Orthogonal Block Transform

Orthogonal Block Transforms

- Linear transform: $\boldsymbol{t} = \boldsymbol{A} \cdot \boldsymbol{s}$ (matrix multiplication)
  $\boldsymbol{s} = \boldsymbol{A}^{-1} \cdot \boldsymbol{t}$ (inverse transform)
- Orthogonal transform: $\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$ (rotation and reflection in signal space)
- Separable transform: Transform of rows and columns of a block

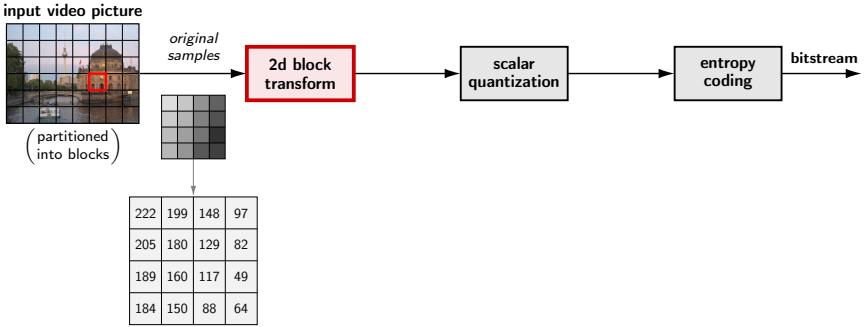
Transforms in Data Compression

- Most often: **Discrete Cosine Transform (DCT)** or approximation thereof

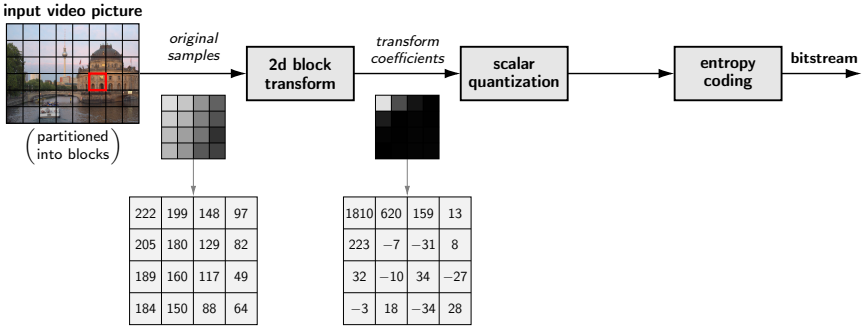


➜ **Effect of Transform: Compaction of Signal Energy**

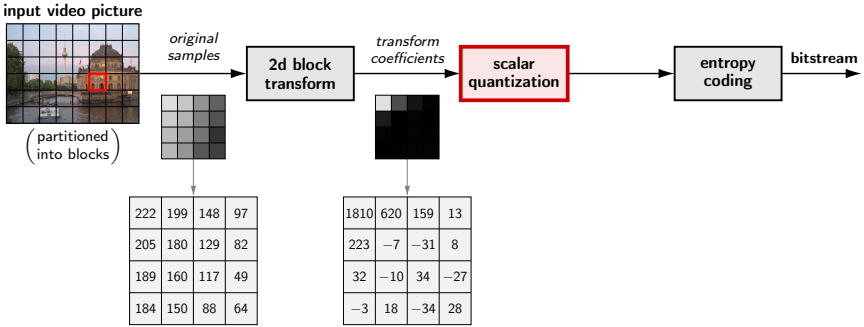
JPEG Principle — Transform Coding of Sample Blocks



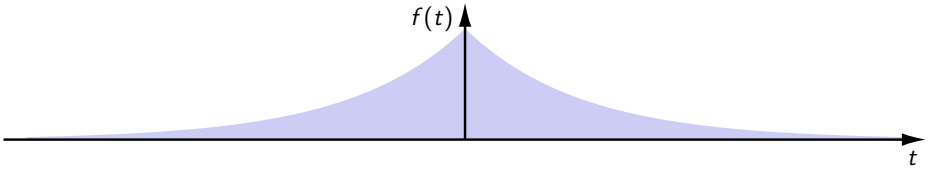
JPEG Principle — Transform Coding of Sample Blocks



JPEG Principle — Transform Coding of Sample Blocks

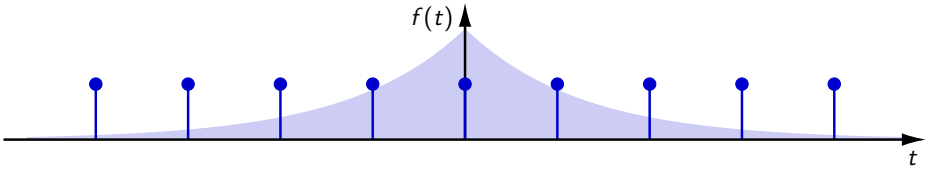


Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

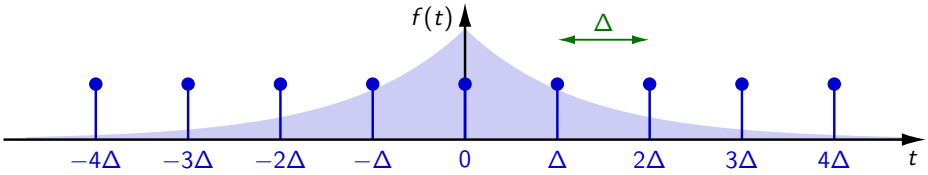
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)

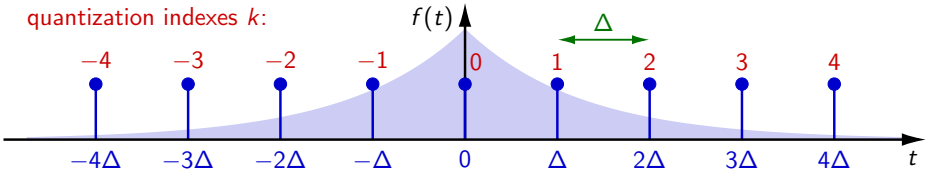
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)

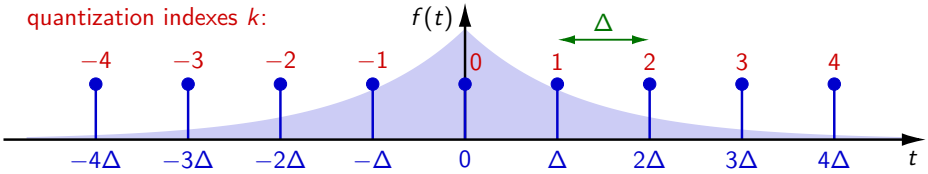
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)

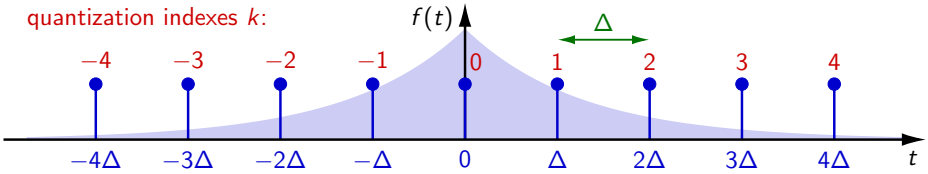
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)
- **Simple decoder operation**: $t' = k \cdot \Delta$ (k: quantization index)

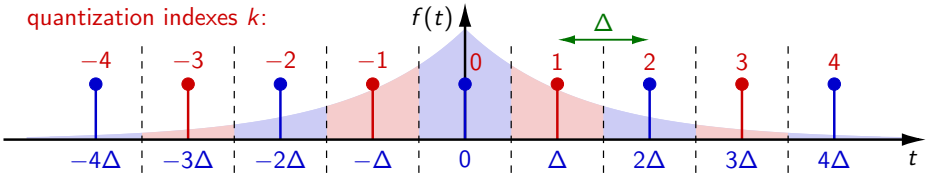
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)
- **Simple decoder operation**: $t' = k \cdot \Delta$ (k: quantization index)
- **Encoder**: Freedom to adapt decision to source and entropy coding

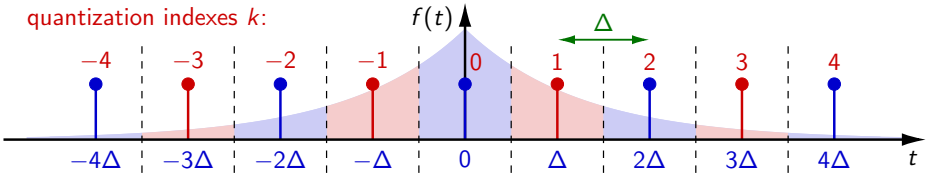
Scalar Quantization



Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)
- **Simple decoder operation**: $t' = k \cdot \Delta$ (k: quantization index)
- **Encoder**: Freedom to adapt decision to source and entropy coding
  - ➡ Simplest encoder: $k = \mathrm{round}\,(t/\Delta)$

Scalar Quantization

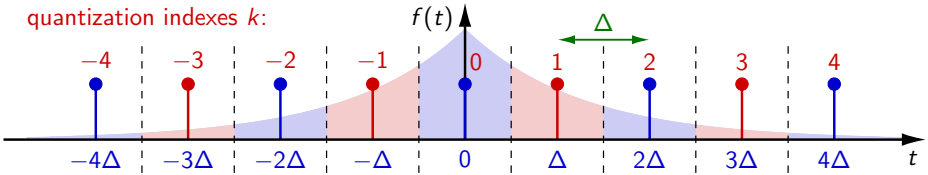


Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)
- **Simple decoder operation**: $t' = k \cdot \Delta$ (k: quantization index)
- **Encoder**: Freedom to adapt decision to source and entropy coding
  - ➡ Simplest encoder: $k = \mathrm{round}\,(t/\Delta)$

➡ **Effect of Quantization: Approximation of Original Signal**

Scalar Quantization



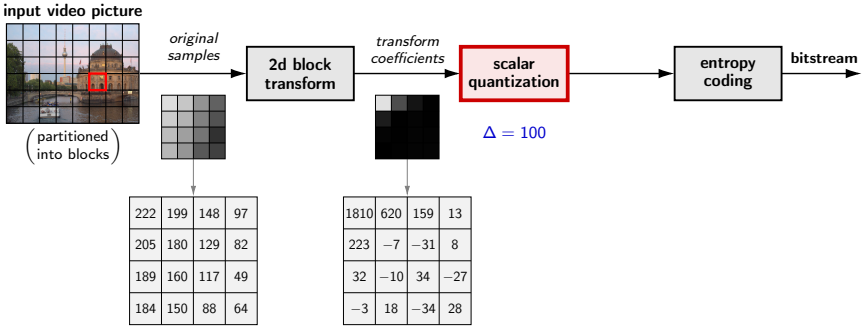
Typical Scalar Quantizer: Uniform Reconstruction Quantizer

- Reconstruction levels: Uniformly spaced and centered around zero (quantization step size Δ)
- **Simple decoder operation**: $t' = k \cdot \Delta$ (k: quantization index)
- **Encoder**: Freedom to adapt decision to source and entropy coding
  - ➡ Simplest encoder: $k = \mathrm{round}\,(t/\Delta)$

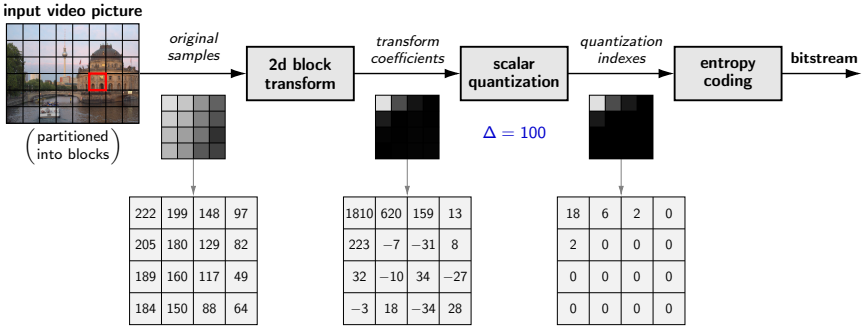
➡ **Effect of Quantization: Approximation of Original Signal**

- ➡ Quantization step size Δ determines trade-off between quality and bit rate

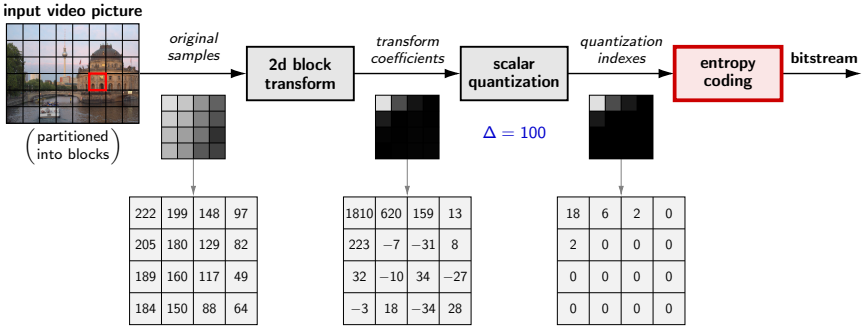
JPEG Principle — Transform Coding of Sample Blocks



JPEG Principle — Transform Coding of Sample Blocks



JPEG Principle — Transform Coding of Sample Blocks



Entropy Coding of Quantization Indexes

Lossless Coding of Quantization Indexes

Entropy Coding of Quantization Indexes

Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code

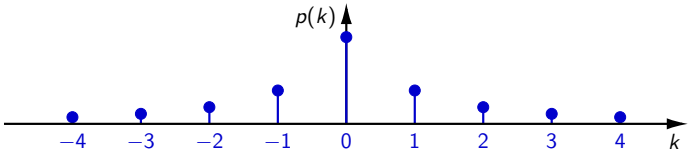
Entropy Coding of Quantization Indexes

Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code

k	simple codewords
0	000000
±1	000001 s
±2	000010 s
±3	000011 s
±4	000100 s
±5	000101 s
±6	000110 s
±7	000111 s
±8	001000 s
...	...
±14	001110 s
±15	001111 s
±16	010000 s
±17	010001 s
±18	010010 s
...	...
±63	111111 s

Entropy Coding of Quantization Indexes

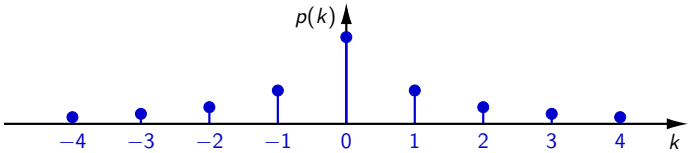


Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code

k	simple codewords
0	000000
±1	000001 s
±2	000010 s
±3	000011 s
±4	000100 s
±5	000101 s
±6	000110 s
±7	000111 s
±8	001000 s
...	...
±14	001110 s
±15	001111 s
±16	010000 s
±17	010001 s
±18	010010 s
...	...
±63	111111 s

Entropy Coding of Quantization Indexes

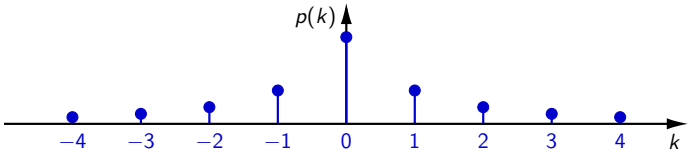


Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code
- Consider symbol probabilities

k	simple codewords
0	000000
±1	000001 s
±2	000010 s
±3	000011 s
±4	000100 s
±5	000101 s
±6	000110 s
±7	000111 s
±8	001000 s
...	...
±14	001110 s
±15	001111 s
±16	010000 s
±17	010001 s
±18	010010 s
...	...
±63	111111 s

Entropy Coding of Quantization Indexes

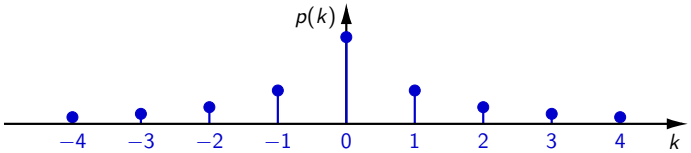


Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code
- Consider symbol probabilities

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

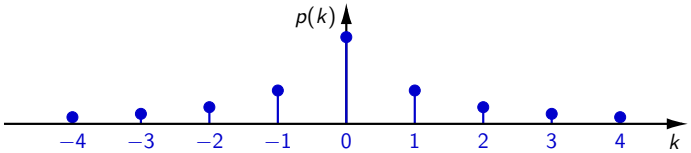
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

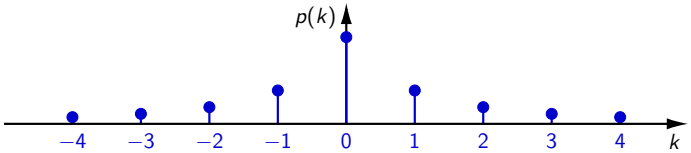
- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)

➡ simple:

➡ better:

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

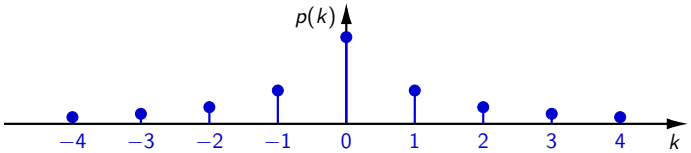
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: 7
- ➡ better: 10

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

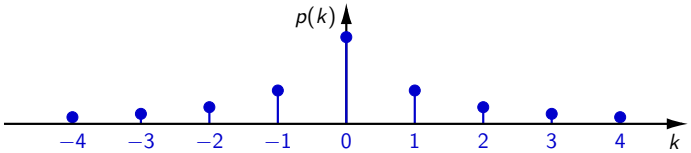
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: $7 + 7$
- ➡ better: $10 + 6$

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

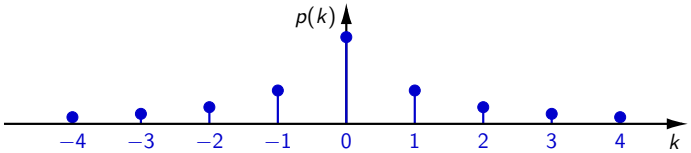
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: $7 + 7 + 2 \cdot 7$
- ➡ better: $10 + 6 + 2 \cdot 4$

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

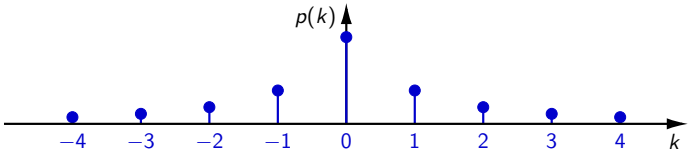
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: $7 + 7 + 2 \cdot 7 + 12 \cdot 6$
- ➡ better: $10 + 6 + 2 \cdot 4 + 12 \cdot 1$

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

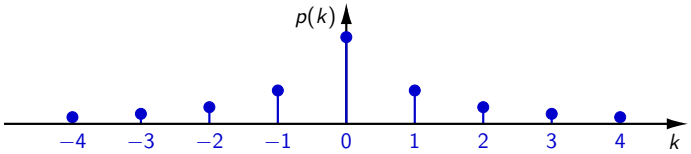
- Simplest approach: Scalar prefix code
- Consider symbol probabilities

Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: $7 + 7 + 2 \cdot 7 + 12 \cdot 6 = 100$ bits
- ➡ better: $10 + 6 + 2 \cdot 4 + 12 \cdot 1 = 36$ bits

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

Entropy Coding of Quantization Indexes



Lossless Coding of Quantization Indexes

- Simplest approach: Scalar prefix code
- Consider symbol probabilities

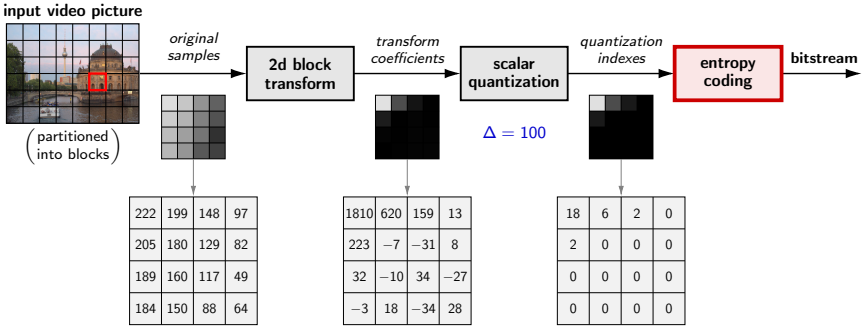
Example

- indexes k: $18, 6, 2, 0, 2, 0, \cdots, 0$ (16 values)
- ➡ simple: $7 + 7 + 2 \cdot 7 + 12 \cdot 6 = 100$ bits
- ➡ better: $10 + 6 + 2 \cdot 4 + 12 \cdot 1 = 36$ bits

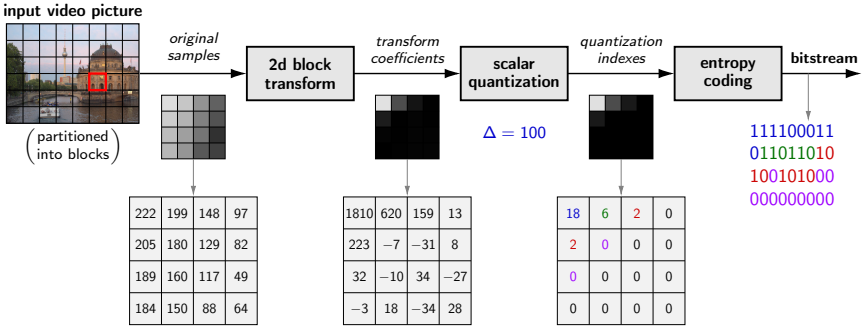
➡ **Entropy Coding: Minimize Number of Bits**

k	simple codewords	better codewords
0	000000	0
±1	000001 s	10 0 s
±2	000010 s	10 1 s
±3	000011 s	110 00 s
±4	000100 s	110 01 s
±5	000101 s	110 10 s
±6	000110 s	110 11 s
±7	000111 s	1110 000 s
±8	001000 s	1110 001 s
...	...	...
±14	001110 s	1110 111 s
±15	001111 s	11110 0000 s
±16	010000 s	11110 0001 s
±17	010001 s	11110 0010 s
±18	010010 s	11110 0011 s
...	...	...
±63	111111 s	1111110 000000 s

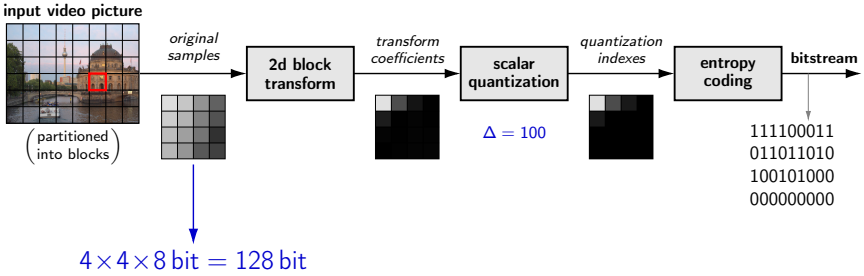
JPEG Principle — Transform Coding of Sample Blocks



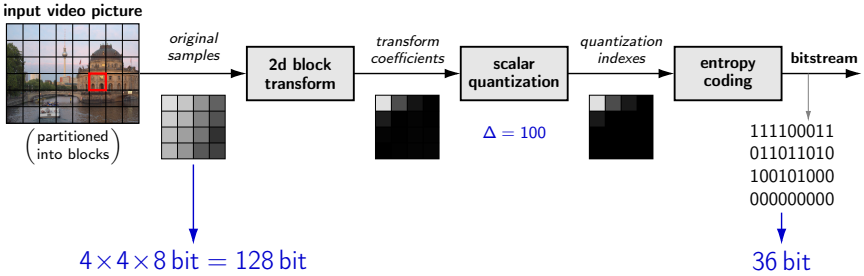
JPEG Principle — Transform Coding of Sample Blocks



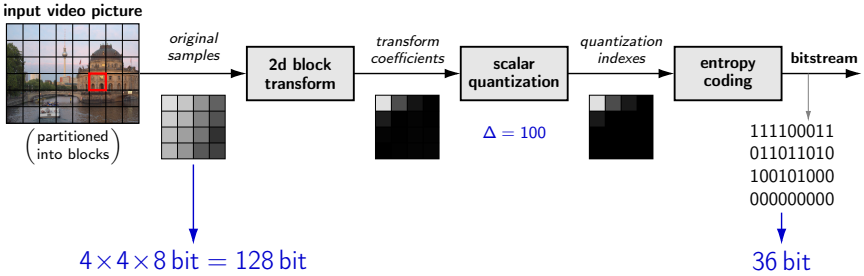
JPEG Principle — Transform Coding of Sample Blocks



JPEG Principle — Transform Coding of Sample Blocks



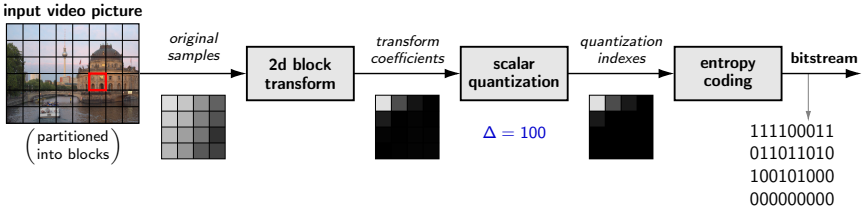
JPEG Principle — Transform Coding of Sample Blocks



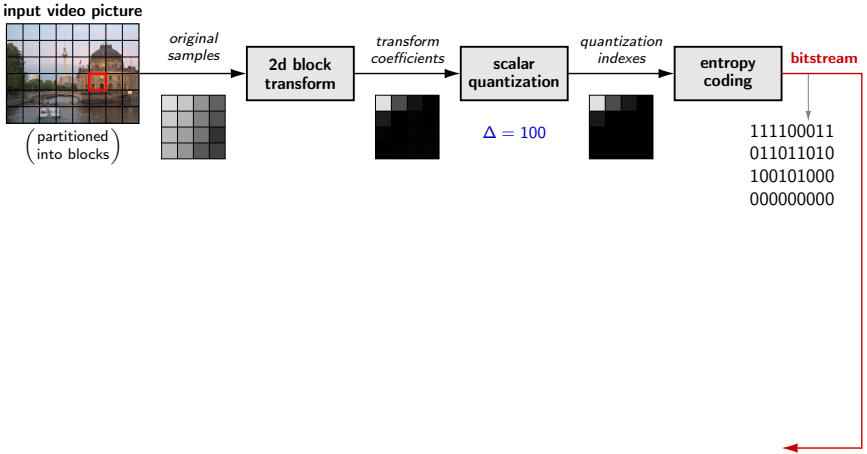
➜ Reduction to 28% of raw data size

➜ Compression factor: 3.6

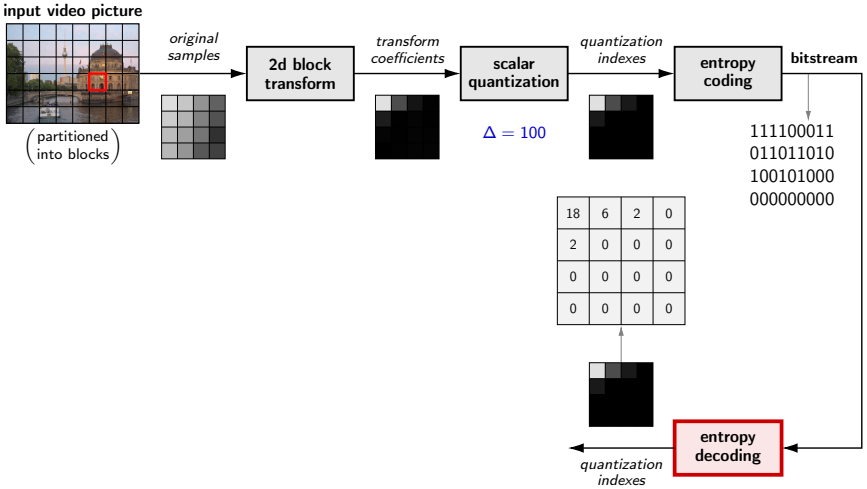
JPEG Principle — Encoding and Decoding of Sample Blocks



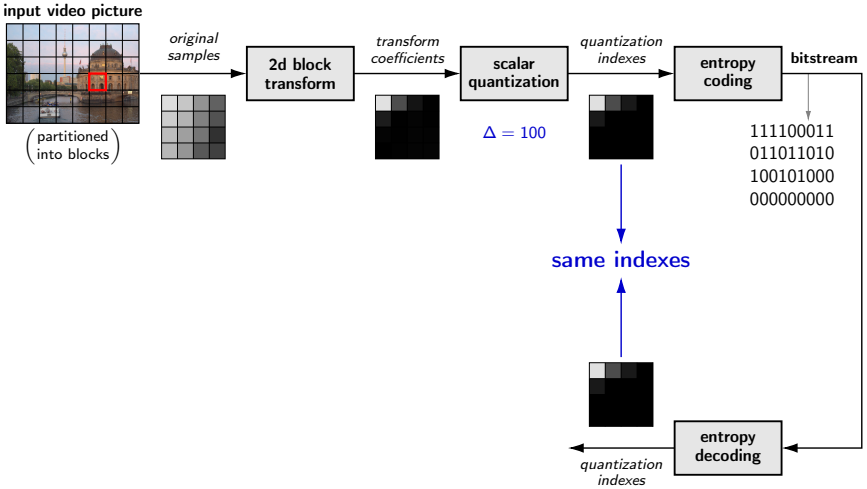
JPEG Principle — Encoding and Decoding of Sample Blocks



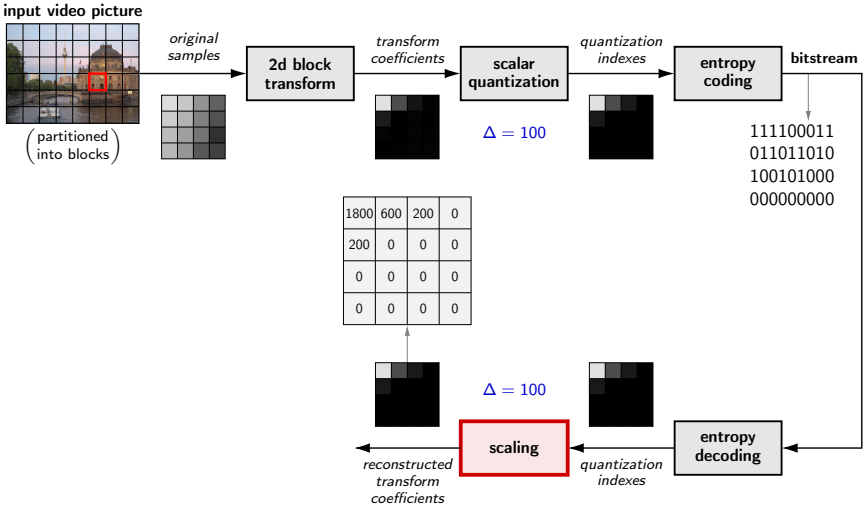
JPEG Principle — Encoding and Decoding of Sample Blocks



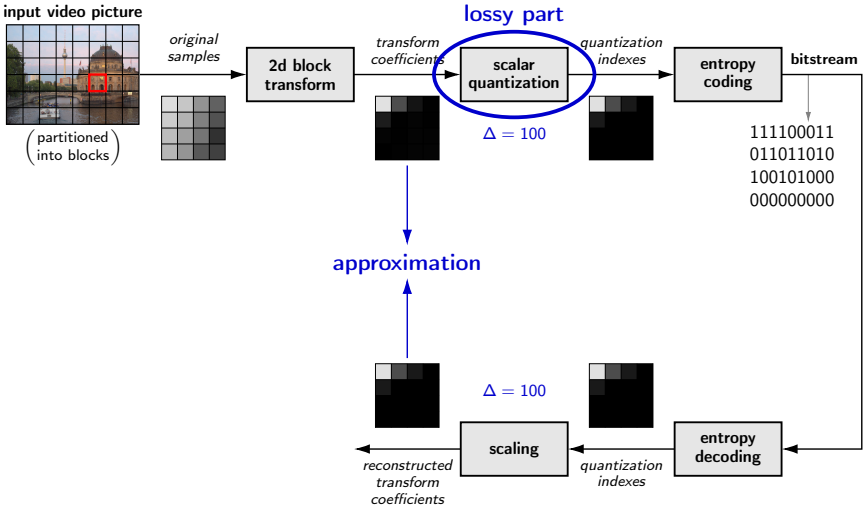
JPEG Principle — Encoding and Decoding of Sample Blocks



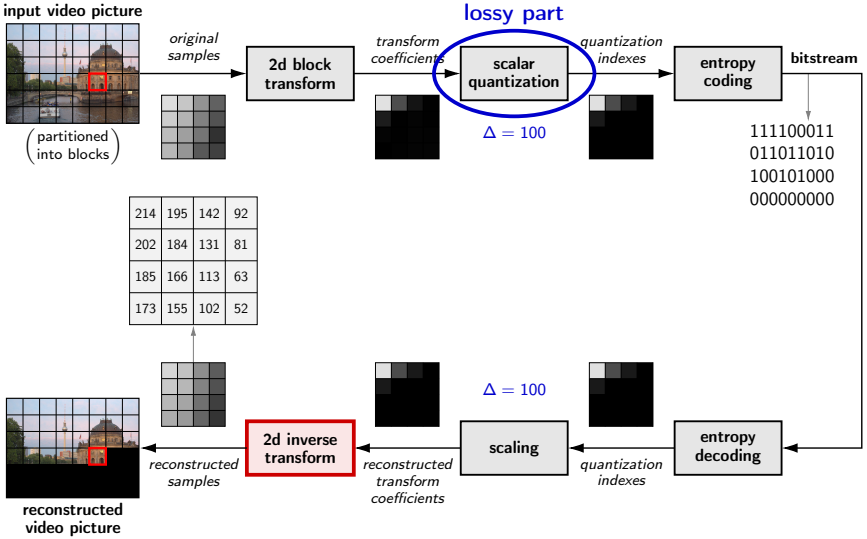
JPEG Principle — Encoding and Decoding of Sample Blocks



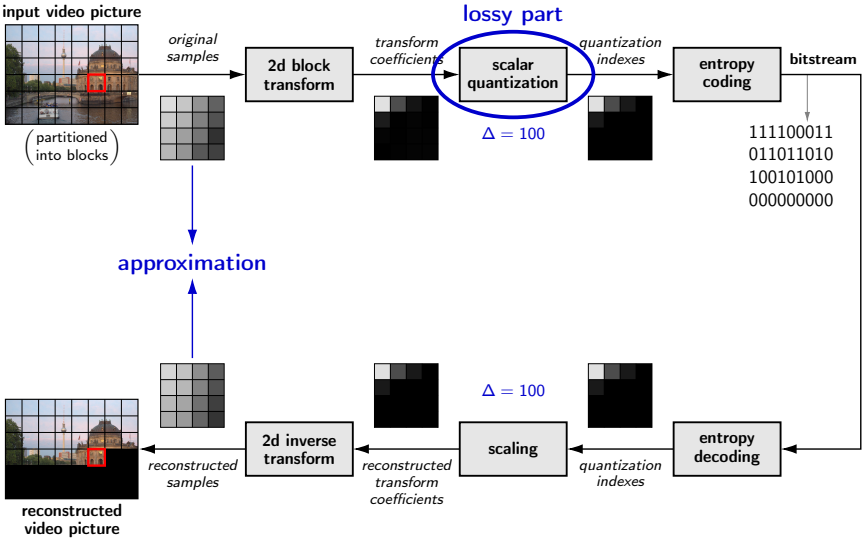
JPEG Principle — Encoding and Decoding of Sample Blocks



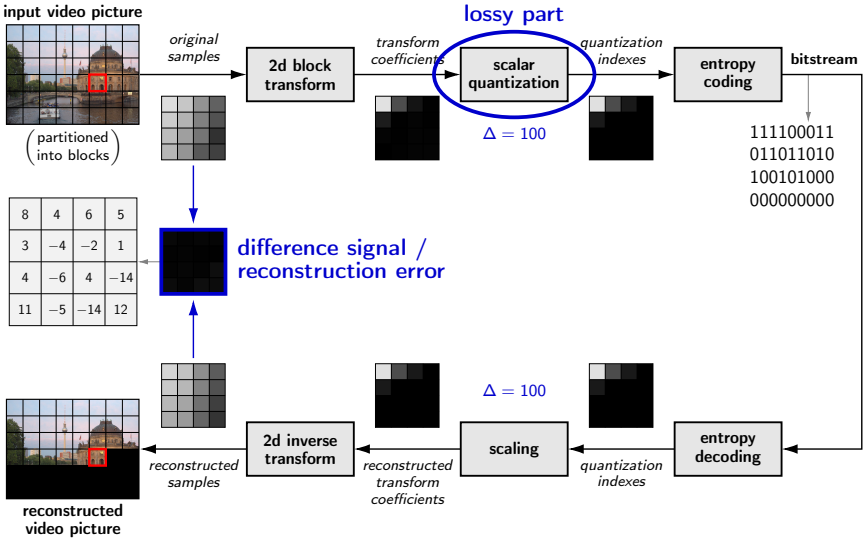
JPEG Principle — Encoding and Decoding of Sample Blocks



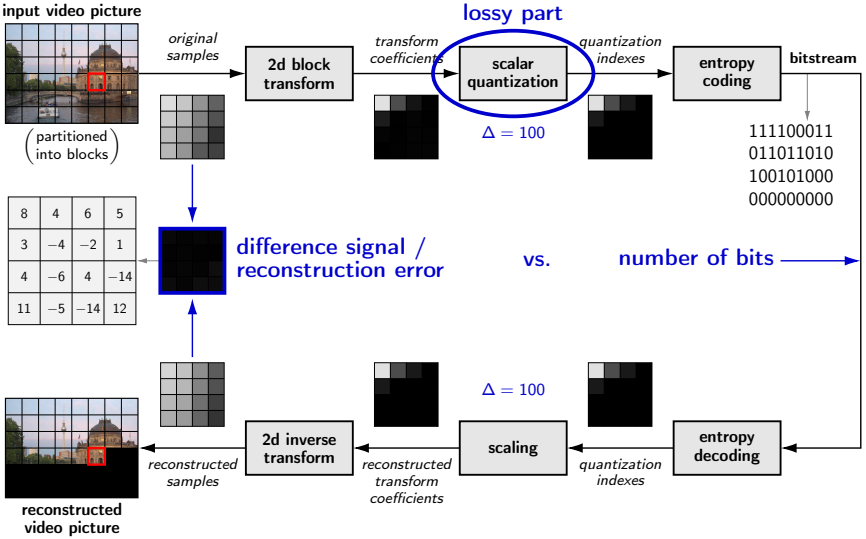
JPEG Principle — Encoding and Decoding of Sample Blocks



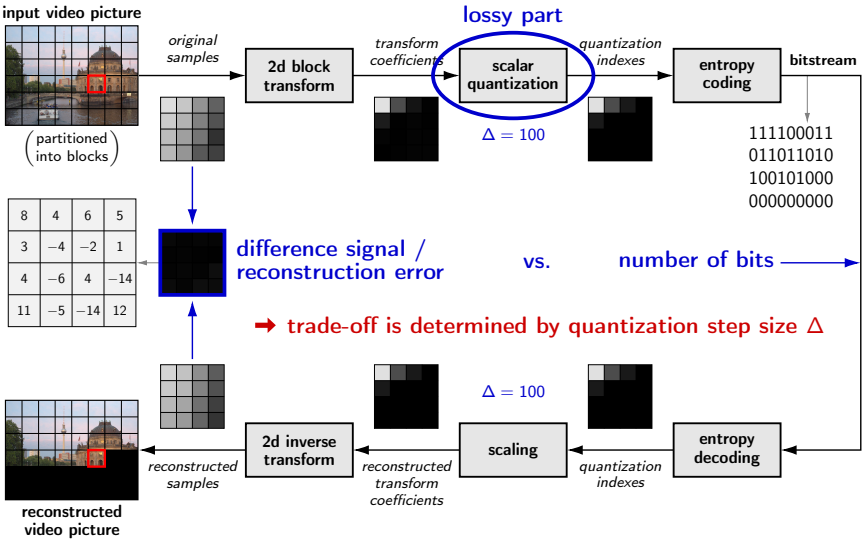
JPEG Principle — Encoding and Decoding of Sample Blocks



JPEG Principle — Encoding and Decoding of Sample Blocks



JPEG Principle — Encoding and Decoding of Sample Blocks



Outline of Course

Lossless Coding

- Unique Decodability and Prefix Codes
- Entropy and Entropy Rate
- Optimal Codes, Huffman Codes
- Arithmetic Coding
- Lempel-Ziv Coding
- Linear Prediction

Lossy Coding

- Scalar Quantization
- Rate-Distortion Functions
- Vector Quantization
- Predictive Quantization
- Transform Coding

Exercise 1: Typical Compression Factors for Media Files

Determine typical compression factors for media files you use every day. Choose one or more examples for the following categories:

- **Images**: Pictures (JPEG) taken with your smartphone
- **Audio**: Songs (MP3 or AAC) you listen
- **Video**: Videos (AVC or ...) you captured or downloaded

Determine the compression factors by

- Measuring the file size
- Calculating the raw data rate, based on
  - for images: image size, color format, and bits per sample
  - for audio: duration, sampling rate, and bit depth
  - for video: duration, frame rate, picture size, color format, bits per sample

Exercise 2: Compare Lossy and Lossless Image Compression

1. Choose one or more of the raw image files (PPM format) provided at the course web site: http://iphome.hhi.de/schwarz/DC.htm
2. Compress the file(s) with a general lossless compression tool (such as zip, rar, ...) and measure the compression factor
3. Convert the file(s) into the PNG format and measure the compression factor
4. Convert the file(s) into the JPEG format and measure the compression factor
5. Can you see a difference between lossy and lossless compression?

The conversion into PNG and JPEG can be done with any suitable software. One example is ImageMagick (available for Window, Linux, MacOS):

- Available at https://www.imagemagick.org
- Conversion from PPM to PNG: convert test.ppm test.png
- Conversion from PPM to JPEG: convert test.ppm test.jpg

Exercise 3: Analysis of JPEG Compression

- Choose one or more of the raw image files (PPM format) provided at the course web site: http://iphome.hhi.de/schwarz/DC.htm
- Compress the image using JPEG with varying quality parameter ($Q = 1..100$)

  # with image magick, you can use the following command line

  convert -quality (Q) test.ppm test.jpg
- What effect has the quality parameter on
  - compression factor / file size
  - reconstruction quality
- Up to which compression factor
  - you cannot distinguish the compressed and the original image
  - does the compressed image looks acceptable
- What kind of compression artefacts do you observe in highly compressed JPEG images?

Exercise 4: Calculus of Probabilities

Repeat the basics of the calculus of probabilities:

- Axiomatic Definition of Probability
- Conditional Probability
- Independence of Events
- Random Variables
- Expectation Values