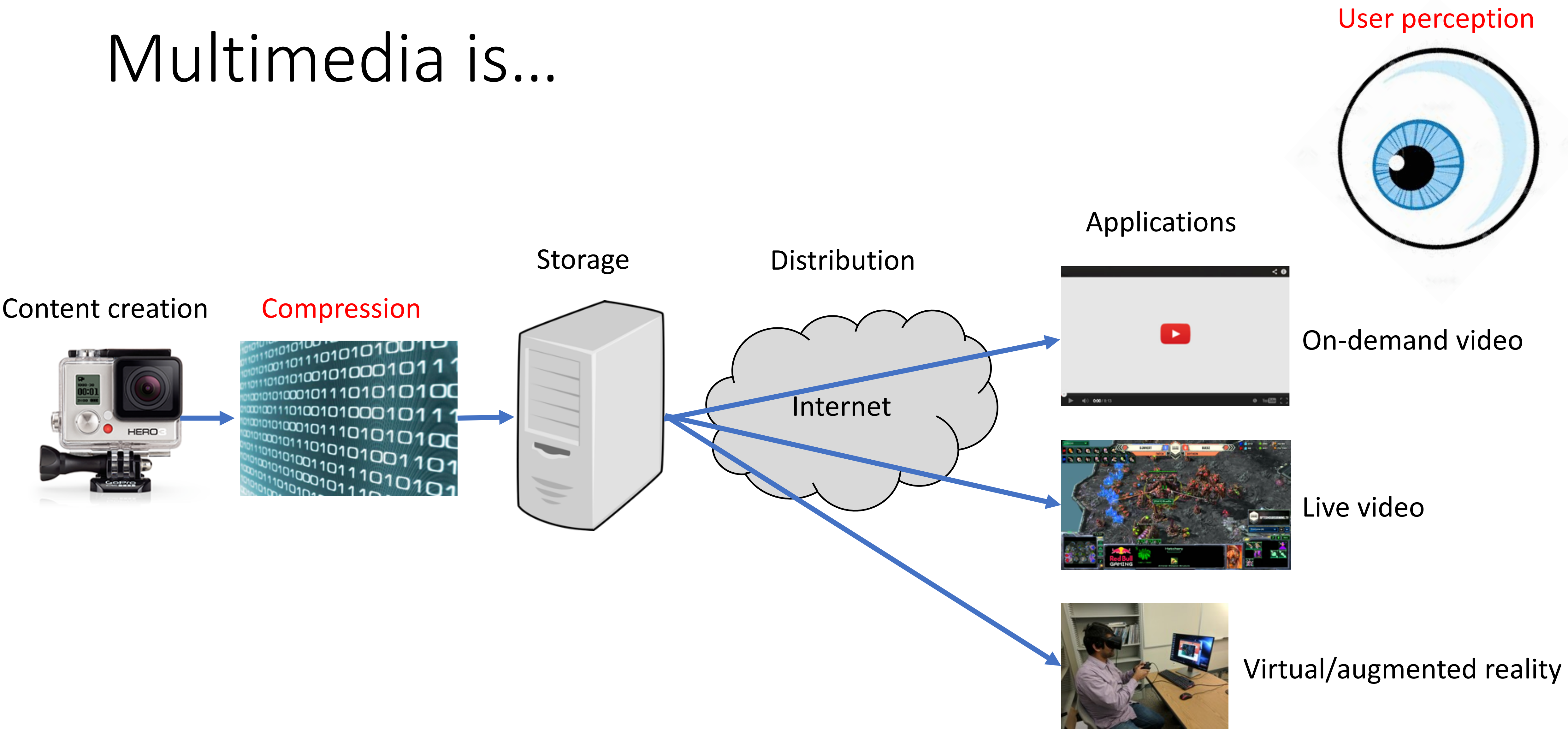
CS 260:
Seminar in Computer Science:
Multimedia Networking

Jiasi Chen

Lectures: MWF 4:10-5pm in CHASS

http://www.cs.ucr.edu/~jiasi/teaching/cs260_spring17/

Multimedia is…



User perception

Applications

Storage

Distribution

Content creation

Compression

Internet

On-demand video

Live video

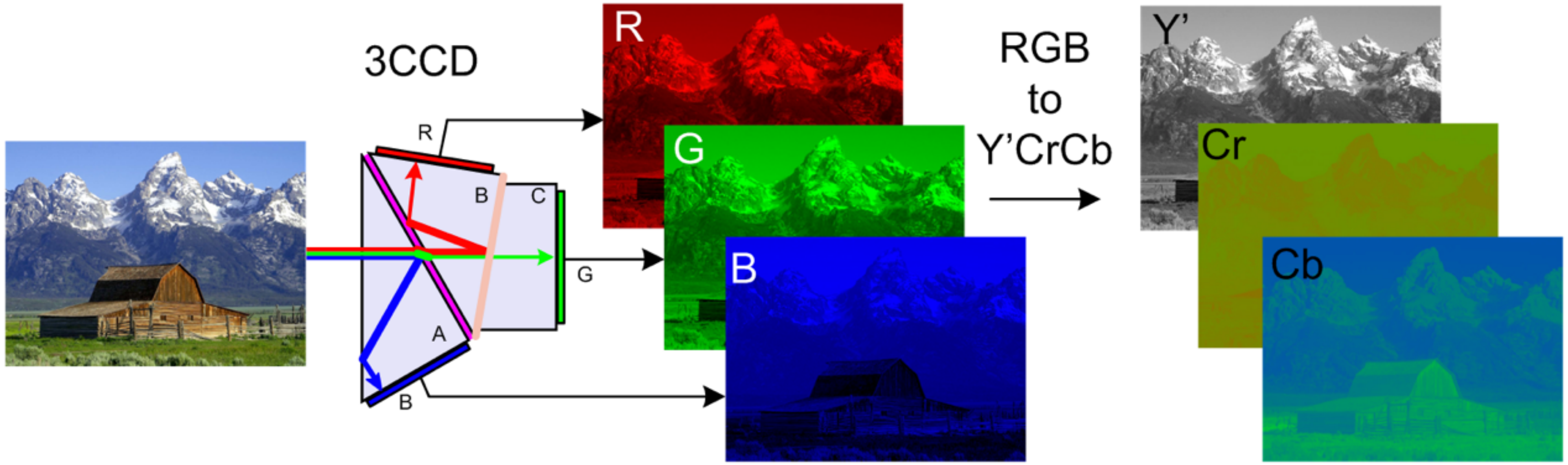
Virtual/augmented reality

Encoding Images

1. Pre-processing
2. Discrete cosine transform
3. Quantization
4. Entropy encoding

Encoding Images: Pre-processing

- Convert from color to luma and chroma components



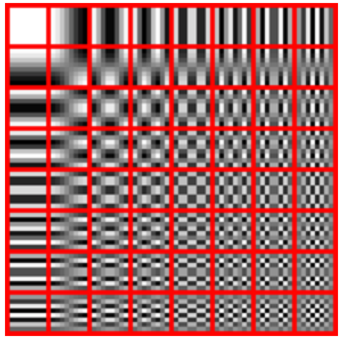
- Divide image into blocks (e.g. 8x8 pixels)

Encoding Images: Discrete Cosine Transform

- Transform from spatial domain to frequency domain

$$g = \overset{\xrightarrow{x}}{\begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix}} \Big\downarrow y. \quad \Longrightarrow \quad G = \overset{\xrightarrow{u}}{\begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.12 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ -48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.87 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix}} \Big\downarrow v.$$

Transformation function $G_{u,v} = \frac{1}{4}\alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} g_{x,y} \cos\left[\frac{(2x+1)u\pi}{16}\right] \cos\left[\frac{(2y+1)v\pi}{16}\right]$ using



basis functions

Example: https://upload.wikimedia.org/wikipedia/commons/5/5e/Idct-animation.gif

Encoding Images: Quantization

- Lossy compression by division and rounding

$$G = \begin{array}{c} \xrightarrow{u} \\ \begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.12 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ -48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.87 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix} \end{array} \Big\downarrow v.$$

$$\Longrightarrow$$

$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

By dividing by

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}.$$

and then rounding.

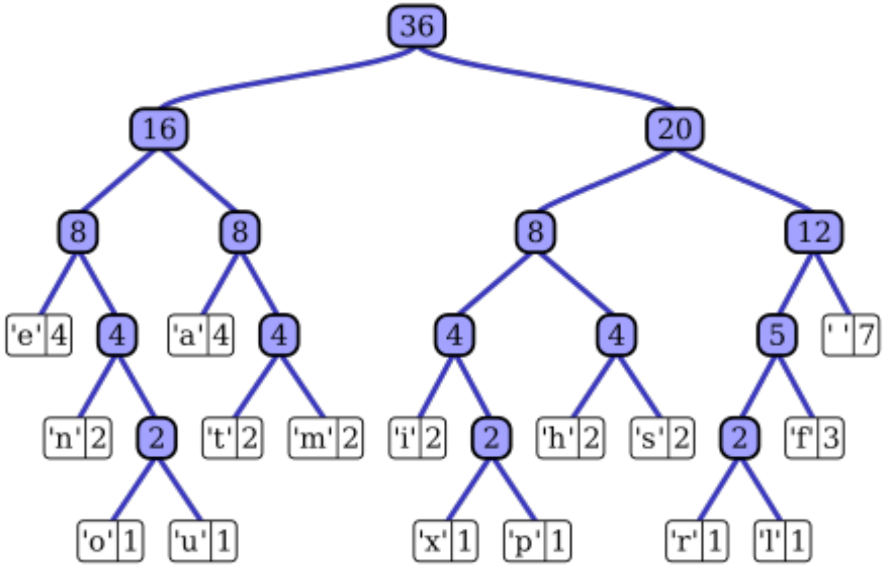
Encoding Images: Entropy Encoding

- Lossless compression to get close to optimal code rate of $-\log_{\#\ \text{symbols}}$(probability of the symbol)

this is an example of a huffman tree →

0110	1010	1000	1011	111	1000 …	135 bits total
t	h	i	s	<space>	i	

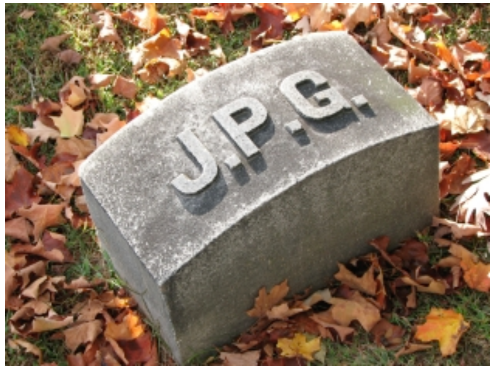
Using the codebook:

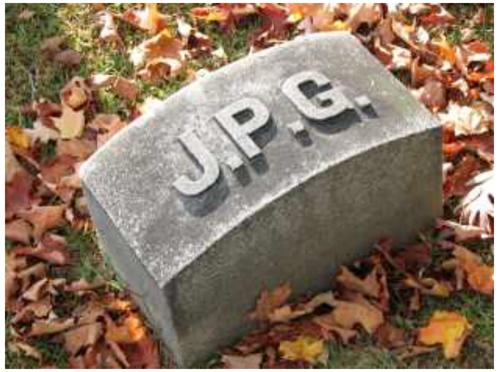


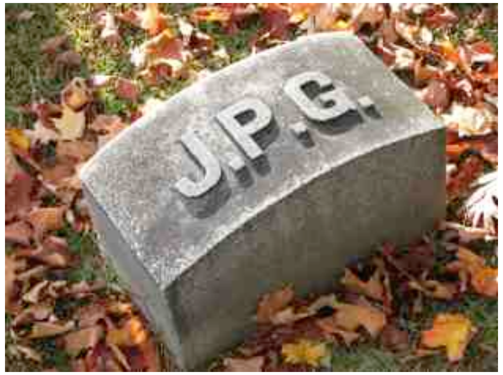
What about the uncompressed version?

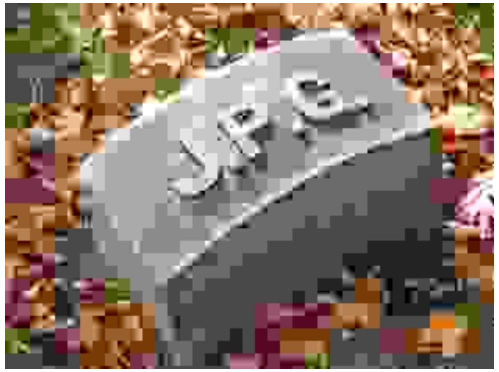
- 26 characters in the alphabet → 5 bits/character
- 5 bits/character * 36 characters in the sentence = 180 bits

Encoding Images: Quality Examples



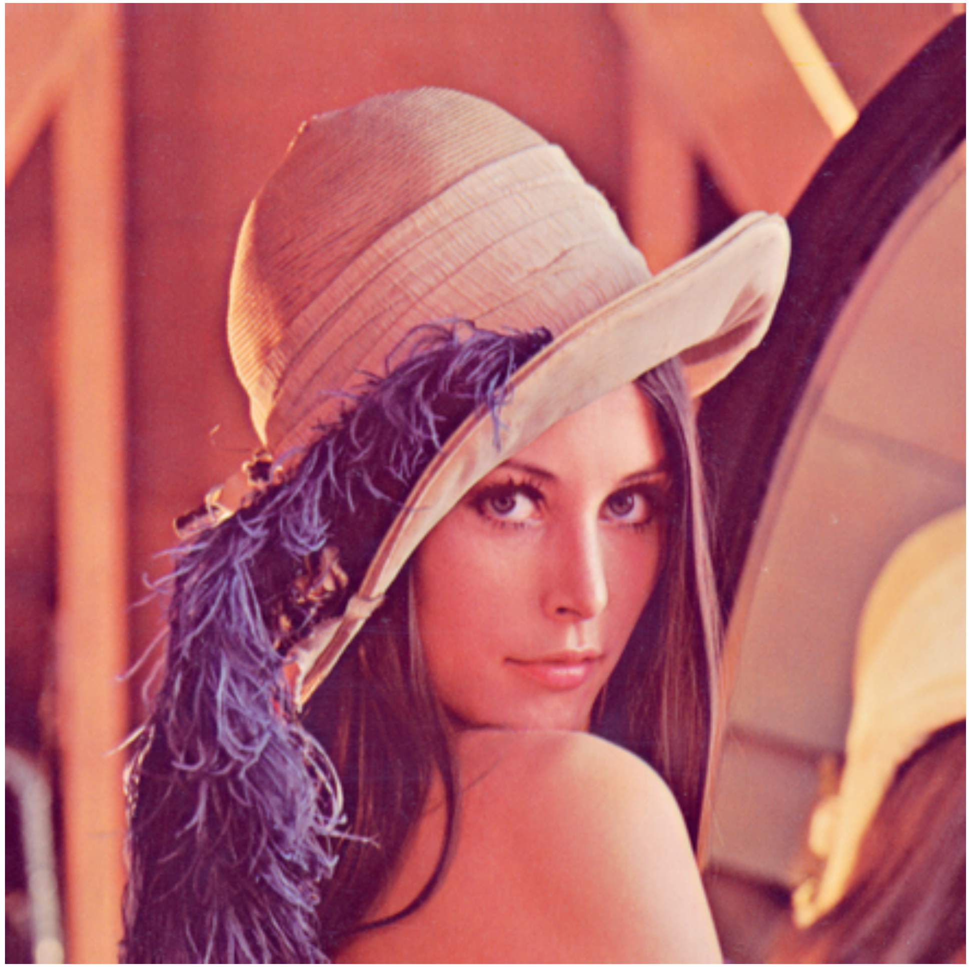






Quality	100	25	10	1
Size	83 bytes	10 bytes	5 bytes	1.5 bytes

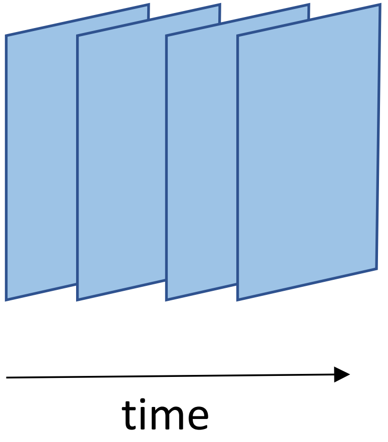
Aside: Lena



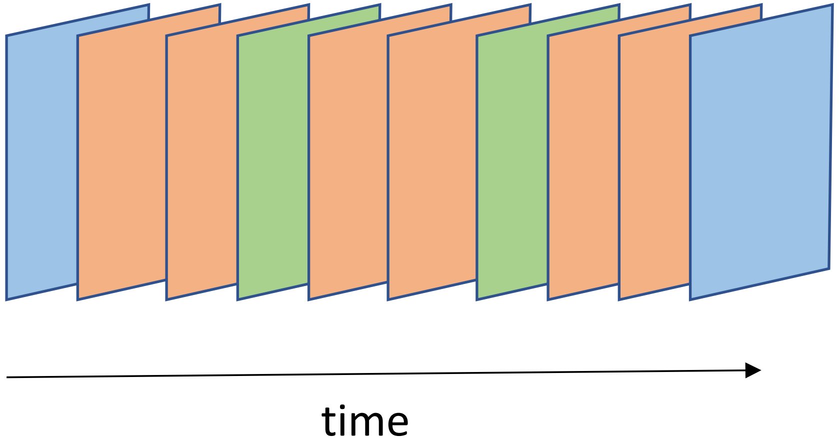
Video Encoding

1. Motion estimation
2. I-frame encoding

Video Encoding: I-frame encoding



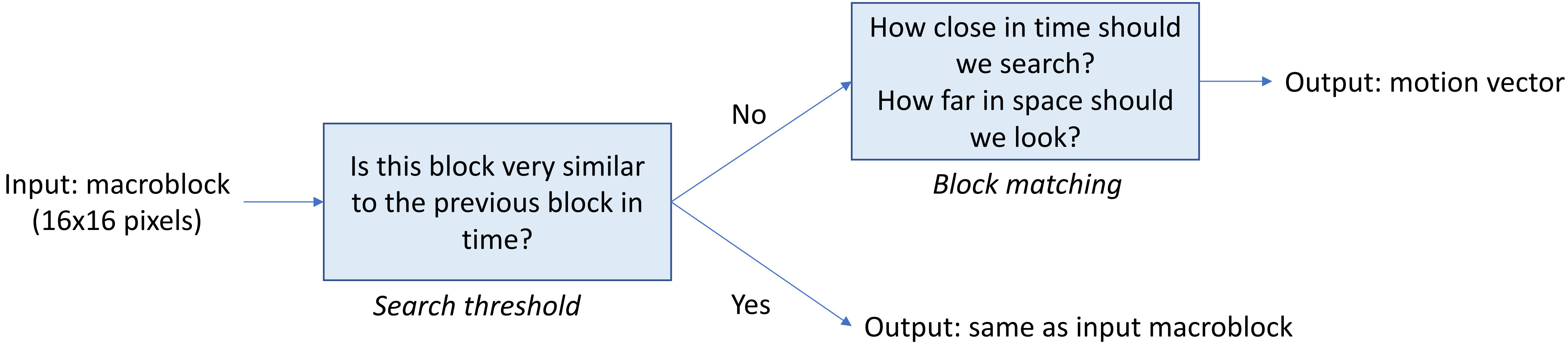
- Naïve solution: encode every frame as a JPEG
- Leverage temporal redundancy by encoding the **difference** between frames
  - I-frame: inter frame
  - P-frame: predictive inter frame
  - B-frame: bi-predictive inter frame



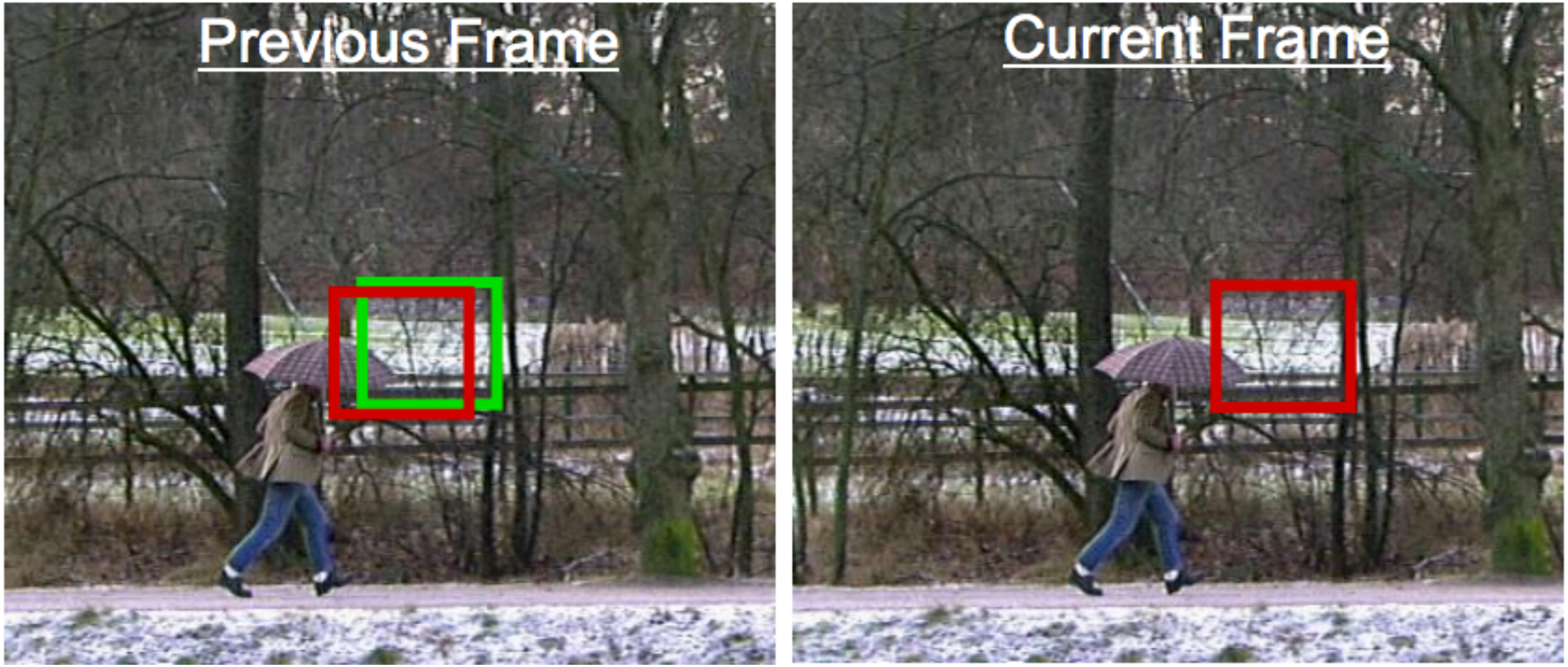
- GOP = “group of pictures” frame pattern
  - E.g., IPPBPPBPP

Video Encoding: Motion Estimation

- How to look for similarity in time?
- Computationally complex

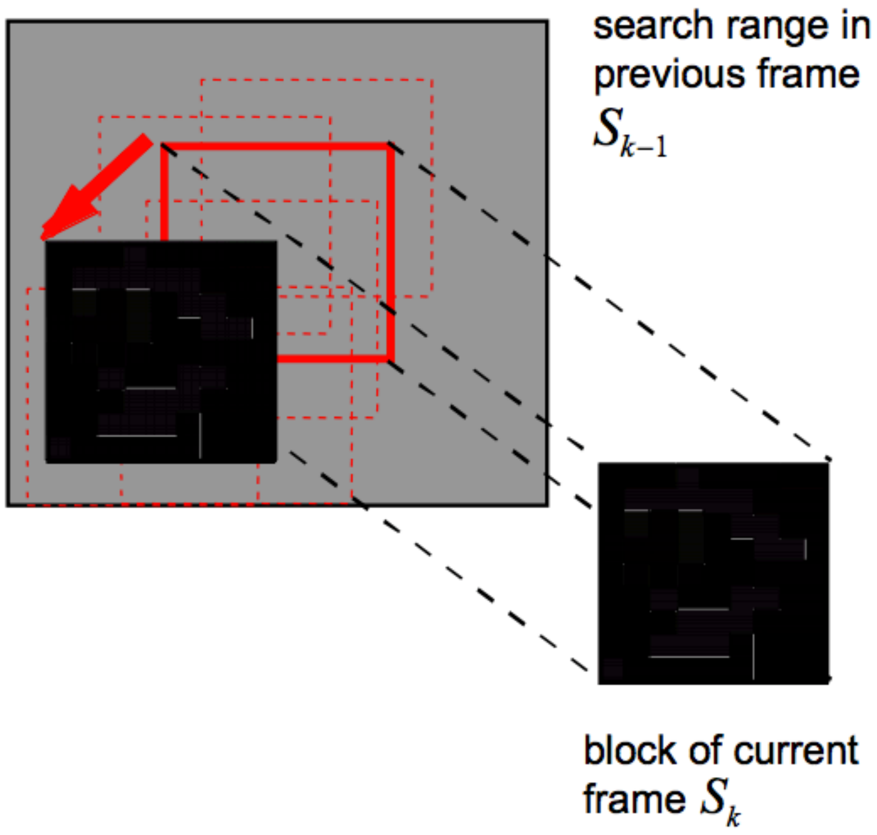

Input: macroblock (16x16 pixels)
Is this block very similar to the previous block in time?
Search threshold
No
Yes
How close in time should we search?
How far in space should we look?
Block matching
Output: motion vector
Output: same as input macroblock

Video Encoding: Block Matching



Source: T. Wiegand / B. Girod: EE398A Image and Video Compression

Video Encoding: Block Matching


search range in previous frame S_{k-1}
block of current frame S_k

- Mean squared error

$$SSD(d_x,d_y)=\sum_{y=1}^{By}\sum_{x=1}^{Bx}[s(x,y,t)-s'(x-d_x,y-d_y,t-\Delta t)]^2$$

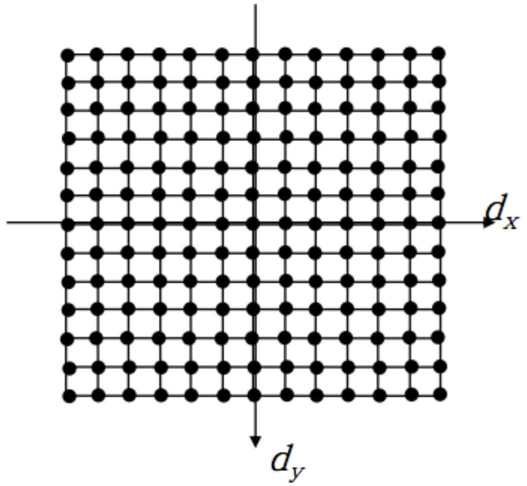
- Sum of absolute differences

$$SAD(d_x,d_y)=\sum_{y=1}^{By}\sum_{x=1}^{Bx}|\,s(x,y,t)-s'(x-d_x,y-d_y,t-\Delta t)\,|$$

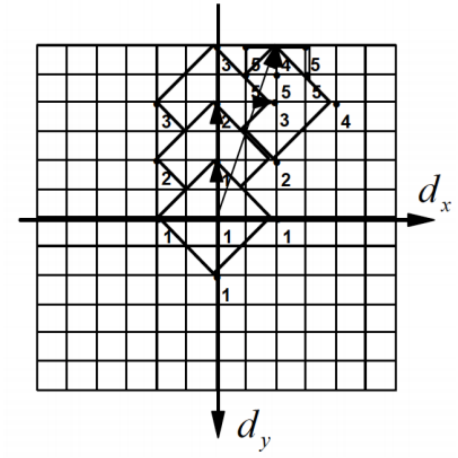
Source: T. Wiegand / B. Girod: EE398A Image and Video Compression

Video Encoding: Search Strategies

Full search



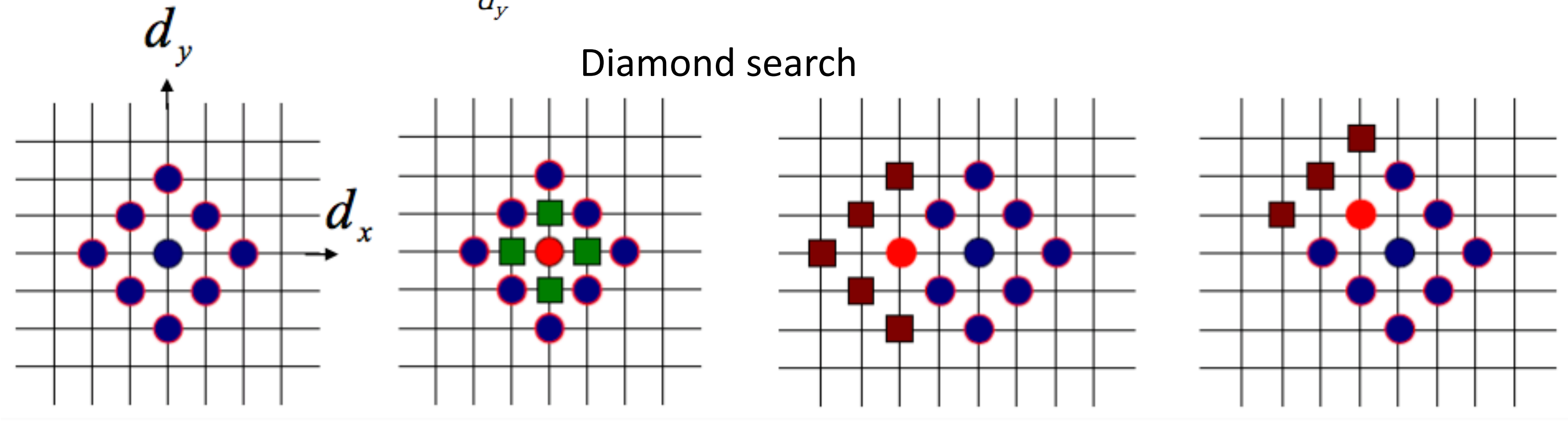
Logarithmic search



General algorithm:

1. Start with an initial step size S
2. Search N locations within S distance
3. If the center is best
   a) S = S/2
   b) Go to 2
4. If an edge location is best
   a) Re-center the origin
   b) Go to 2

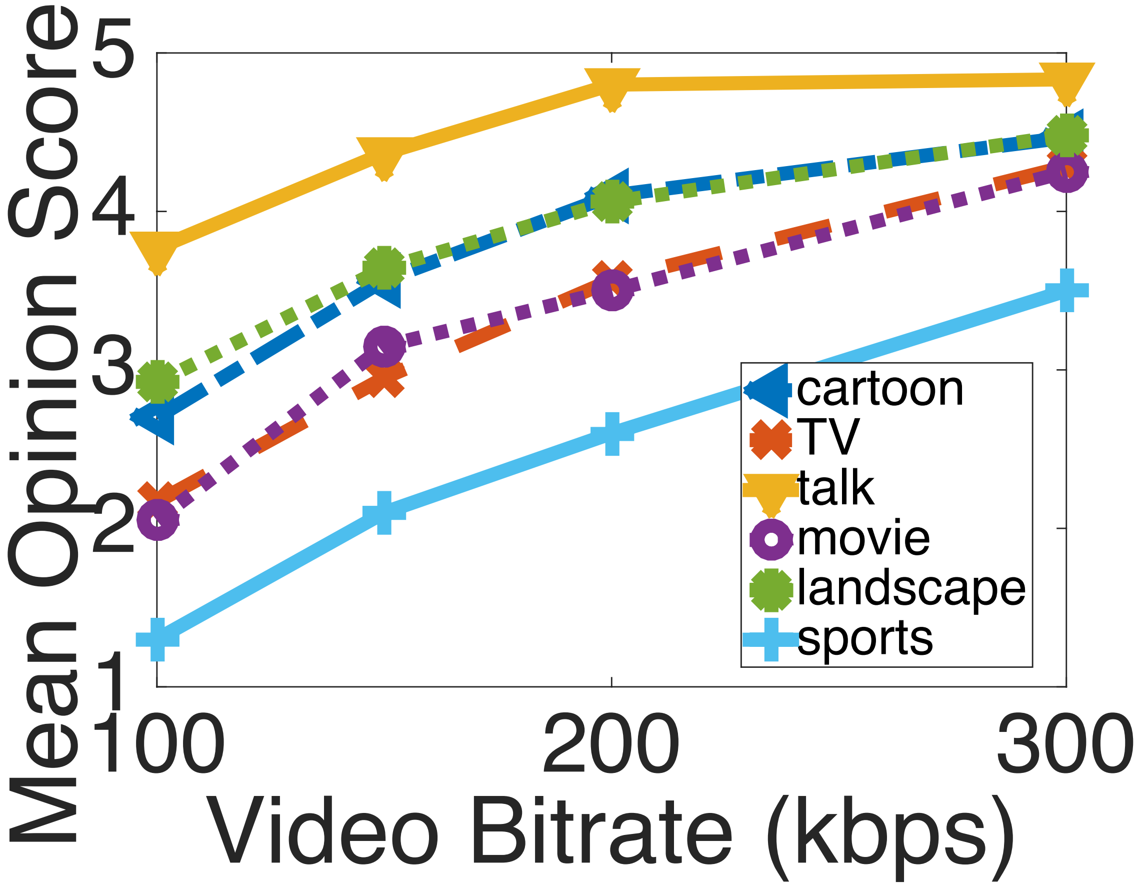
Diamond search



Source: T. Wiegand / B. Girod: EE398A Image and Video Compression

Content Type and Compression

Example: https://www.youtube.com/watch?v=YyRgdWNq-aQ



Video Metrics

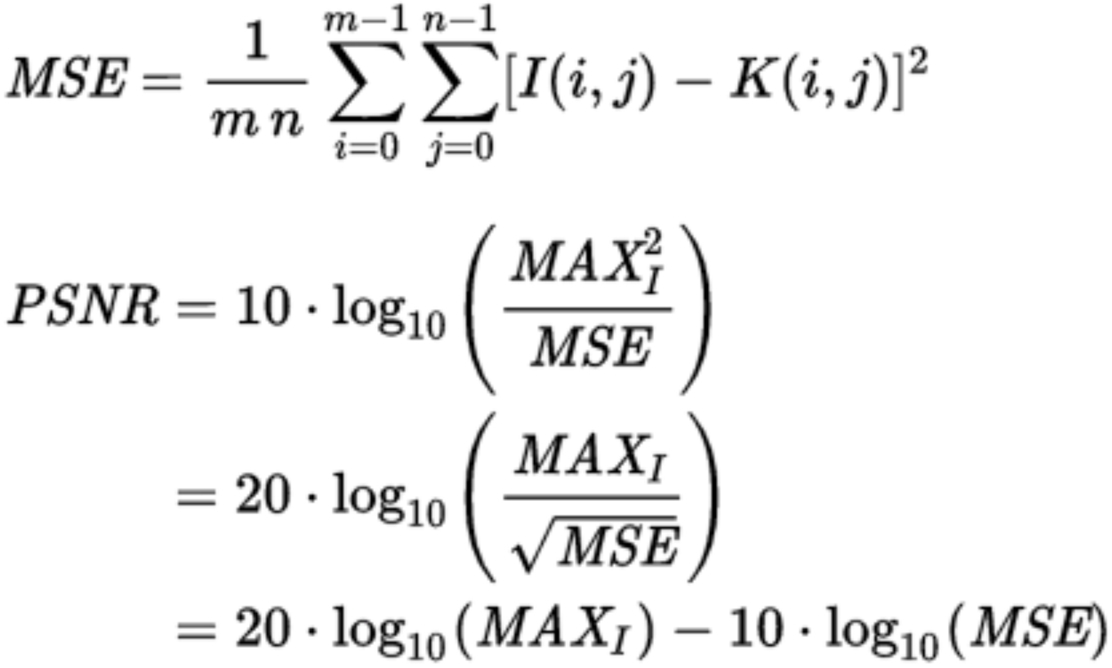
- Resolution = (# pixels) x (# pixels)
  - 720p = 1280 x 720
  - 1080p = 1920 x 1080
  - 4K = 3840 x 2160
- Frames per second
  - 30 fps
  - 60 fps
- Bitrate
  - Wireless: ~1 Mbps
  - Desktop: ~3-5 Mbps
  - High-resolution: 10+ Mbps

- Codec = encoding type
  - H.264
  - VP8
- Container = holds video + audio
  - webm
  - MPEG4

- Decoder
- Encoder

Image Quality: Quantitative Metrics

- How to measure video quality quantitatively?
- PSNR

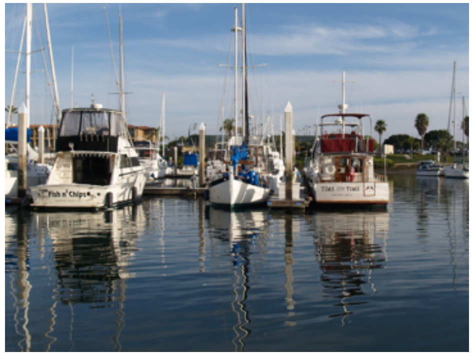


$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

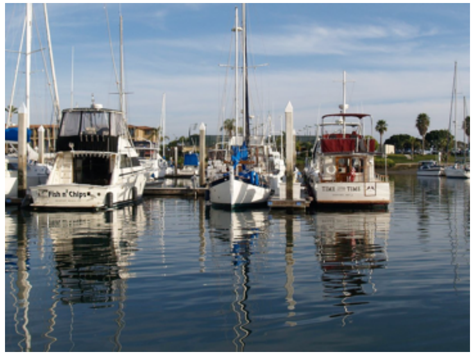
$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ &= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE) \end{aligned}$$

I: original image
K: compressed image
i,j: directions
MAX = max value of pixel

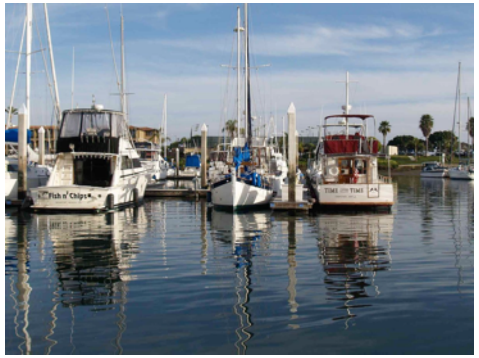
PSNR Example



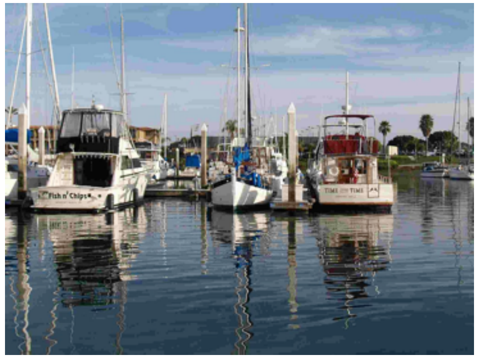
Original uncompressed image



PSNR = 45.53 dB

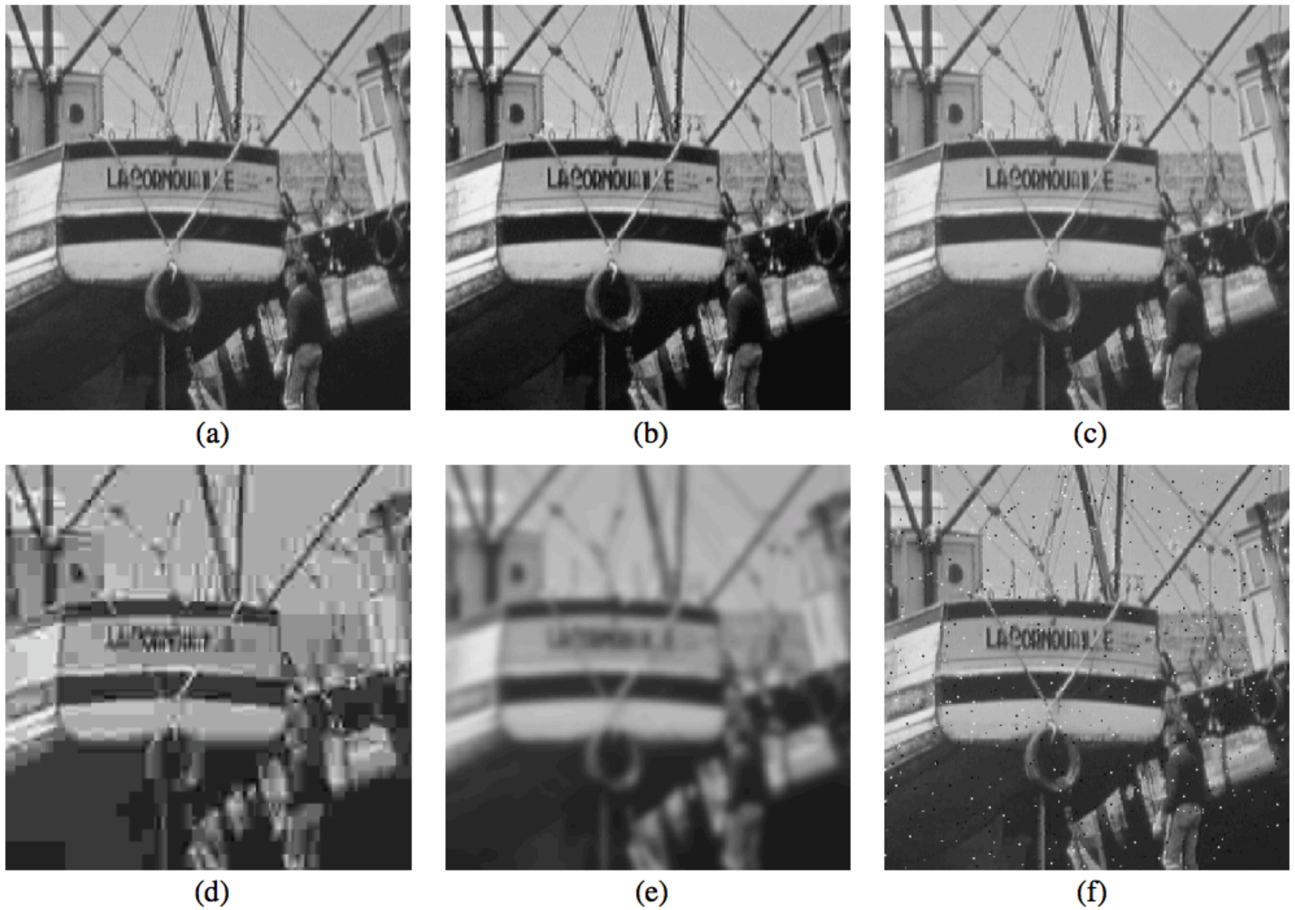


PSNR = 36.81 dB



PSNR = 31.45 dB

Image Quality: Quantitative Metrics



original

increase contrast

mean-shifted

(a)

(b)

(c)

(d)

(e)

(f)

JPEG compression

blur

salt-pepper noise

All of these images have the same MSE

→ Not all errors are created equal

Source: Wang, Zhou; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. (2004-04-01). "Image quality assessment: from error visibility to structural similarity". IEEE Transactions on Image Processing. 13 (4): 600–612.

Video Quality: SSIM

- Key idea: humans are responsive to changes in *structure*
    - E.g., increase contrast or average brightness doesn't matter too much
    - More closely approximate human visual system
    - Operate on luma component only (not color or chrominance)

- Three components
    - Luminance: based on mean $\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i.$
    - Contrast: based on variance, with mean subtracted $\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}$
    - Structure: based on correlation, with mean subtracted and variance normalized

Video Quality: SSIM

- Luminance $l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$
- Contrast $c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$
- Structure $s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$

$$SSIM(x,y) = \left[l(x,y)^{\alpha} \cdot c(x,y)^{\beta} \cdot s(x,y)^{\gamma}\right]$$

α, β, γ = 1, $c_3 = c_2/2$

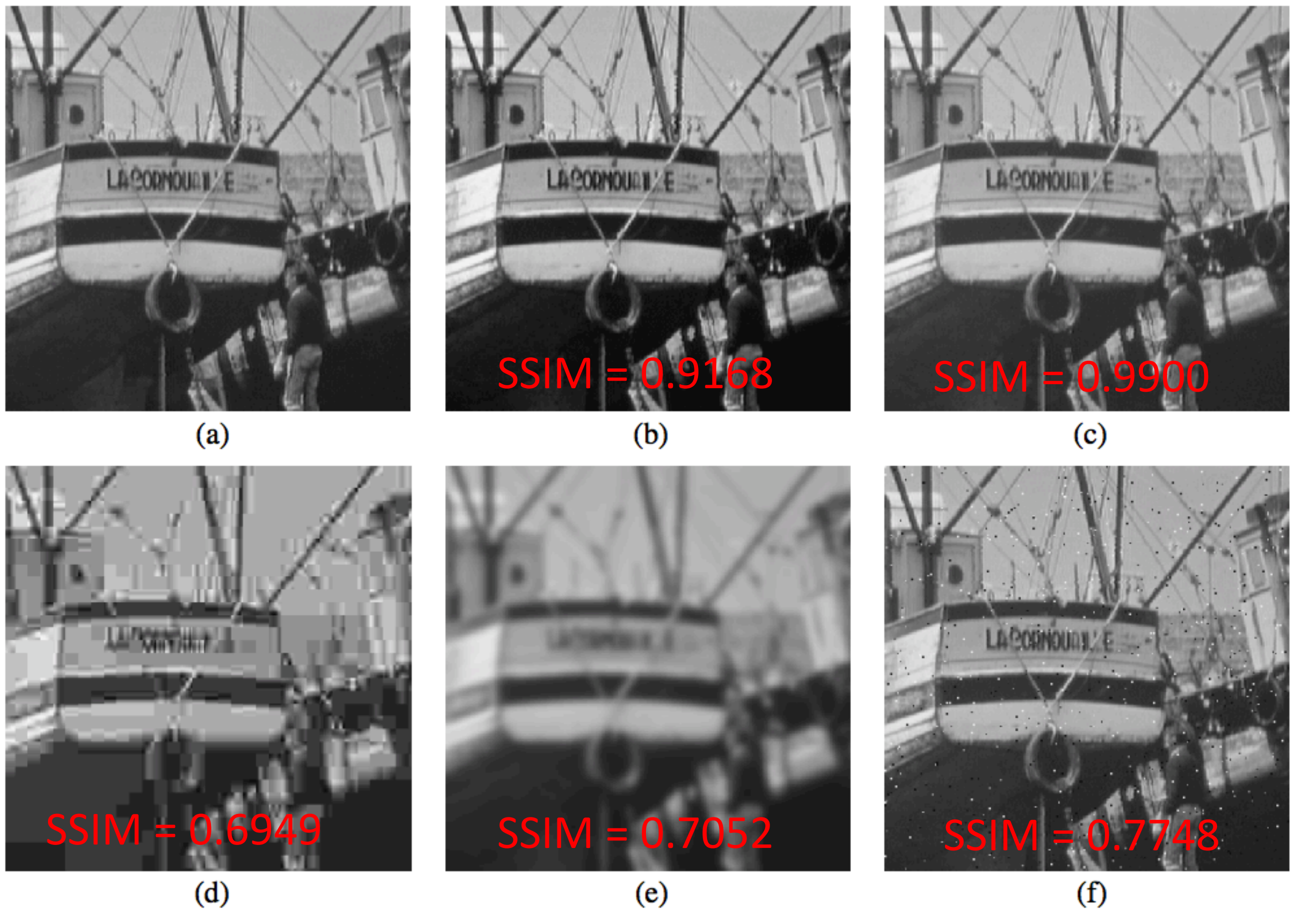
$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Image Quality: Quantitative Metrics

original

increase contrast

mean-shifted



JPEG compression

blur

salt-pepper noise

All of these images have the same MSE = 210

→ Not all errors are created equal

Source: Wang, Zhou; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. (2004-04-01). "Image quality assessment: from error visibility to structural similarity". IEEE Transactions on Image Processing. 13 (4): 600–612.

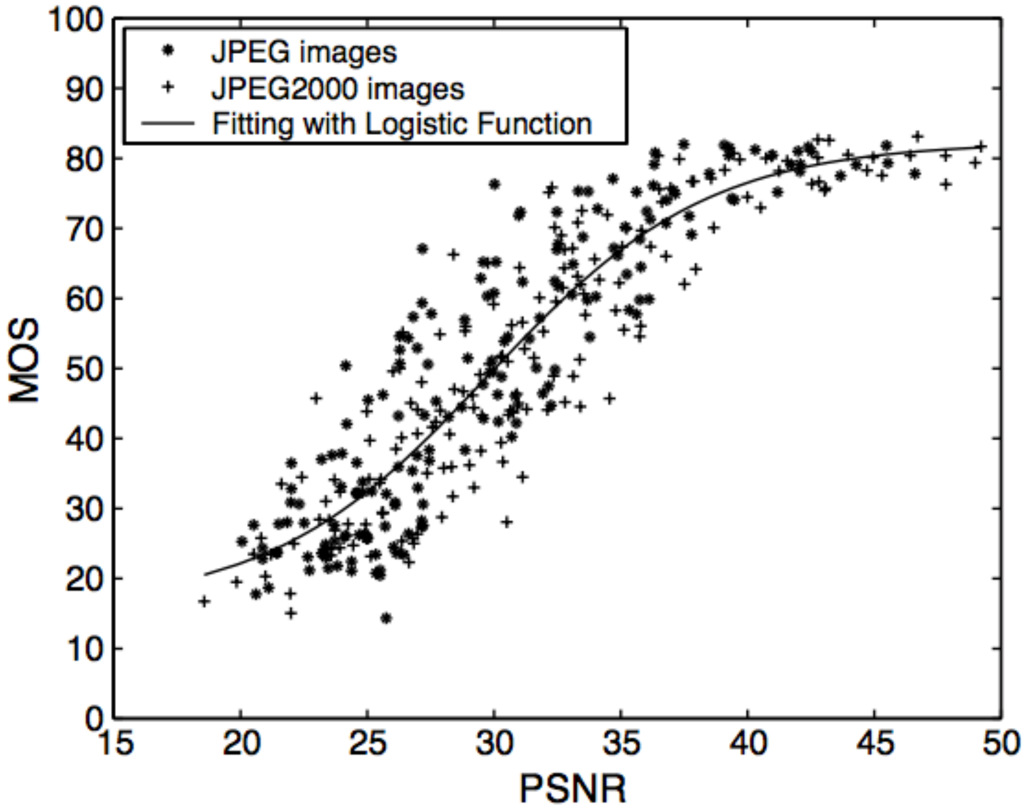
Image Quality: Qualitative Metrics

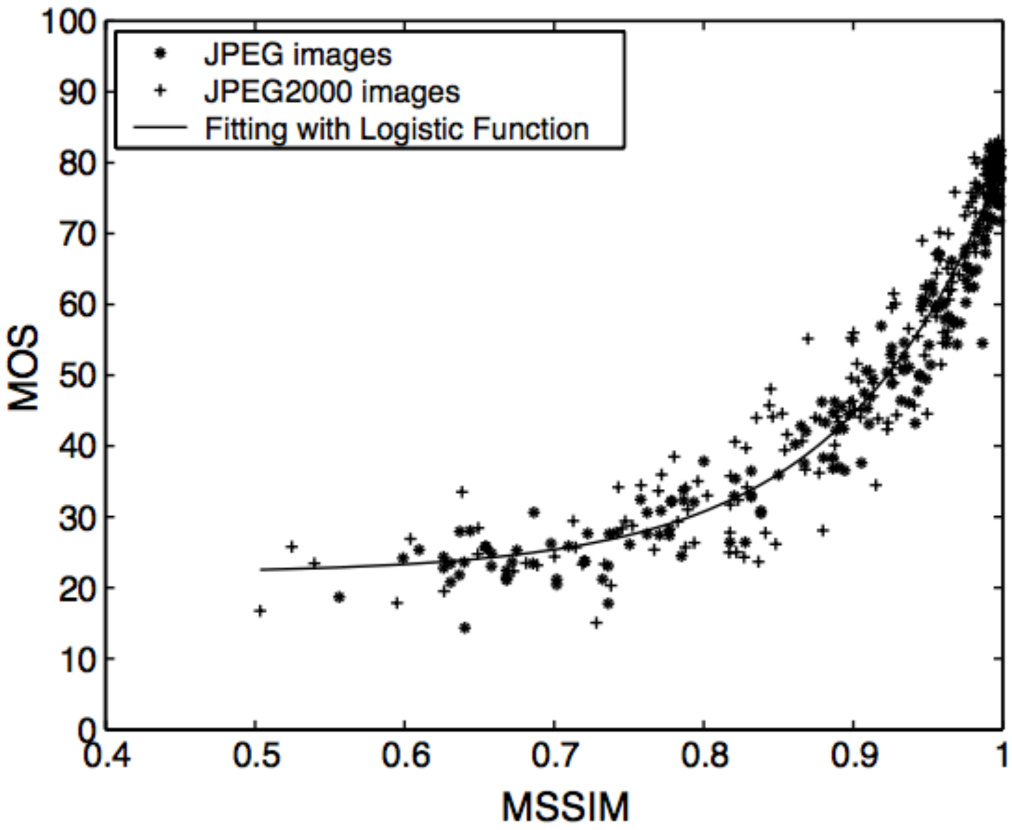
- Mean Opinion Score
  - 5: Excellent
  - 4: Good
  - 3: Fair
  - 2: Poor
  - 1: Bad

- ITU recommendations for how to set up the experiment
  - Distance from viewers, number of views visible, etc.

- User studies can be time-consuming and expensive

Image Quality Metric Comparison





Video Quality

- User quality of experience (QoE)
    - Average PSNR or SSIM across all frames
    - MOS
    - Watch time = how long the user watches the video

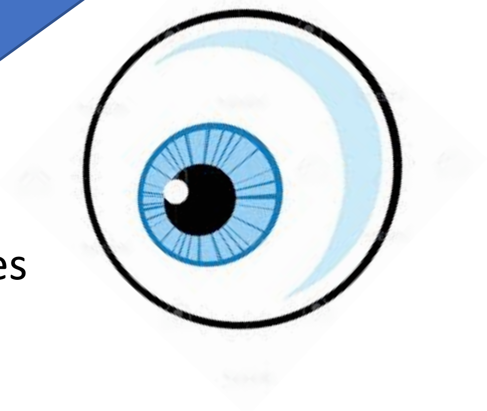
- Video metrics
    - Stalls = # of times the buffer is empty
    - Buffering ratio = # the fraction of time the buffer is empty
    - Bitrate switches = # times the video changes quality
    - Startup time = time from when the user requests the video to when it starts playing

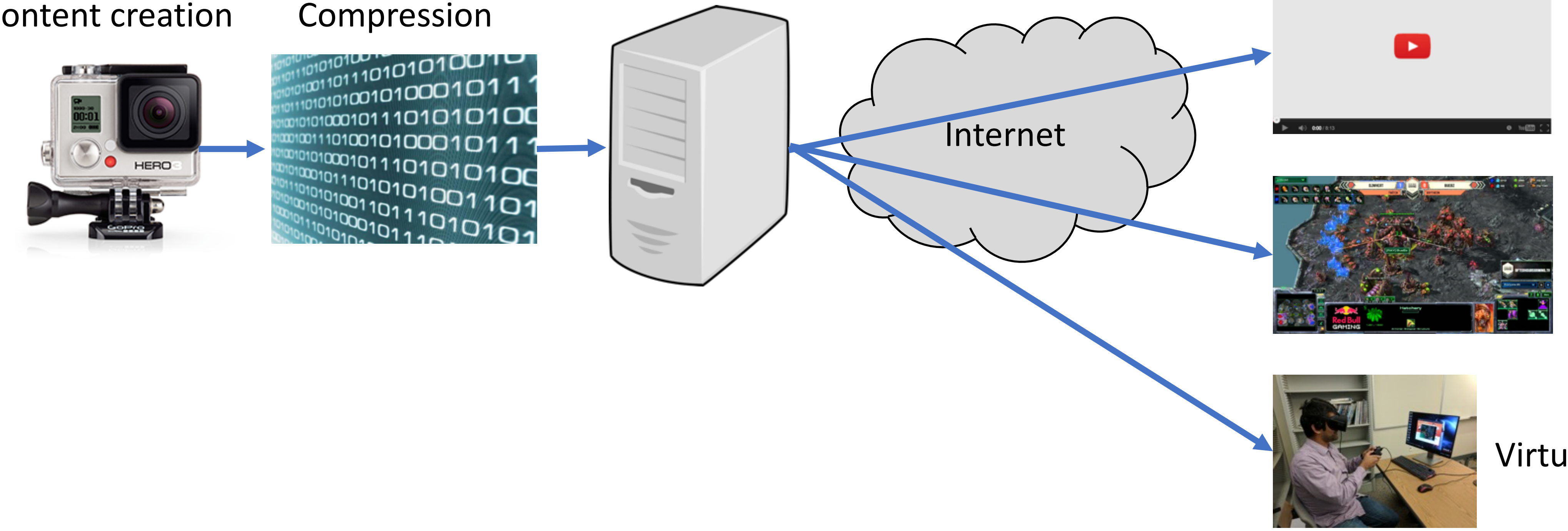
Metrics

Network metrics
- CDN choice
- Throughput
- Latency
- Packet loss

Video metrics
- Stalls
- Buffering ratio
- Bitrate switches
- Startup time

User QoE
- MOS
- PSNR/SSIM





Content creation

Compression

Storage

Distribution

Internet

Applications

On-demand video

Live video

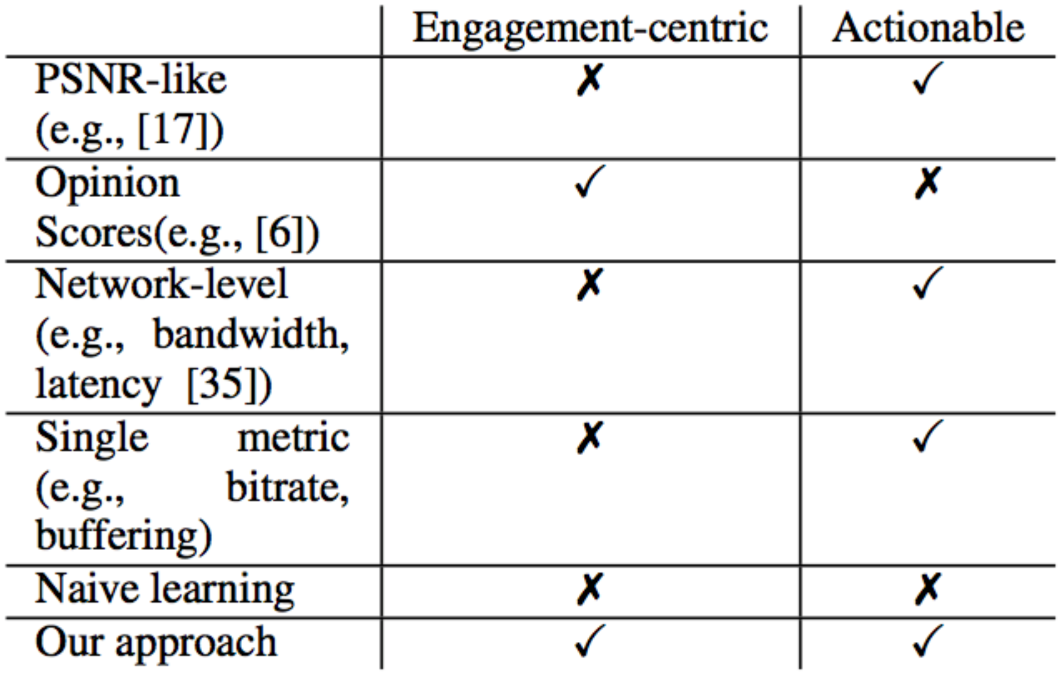
Virtual/augmented reality

Developing a Predictive Model of Quality of Experience for Internet Video

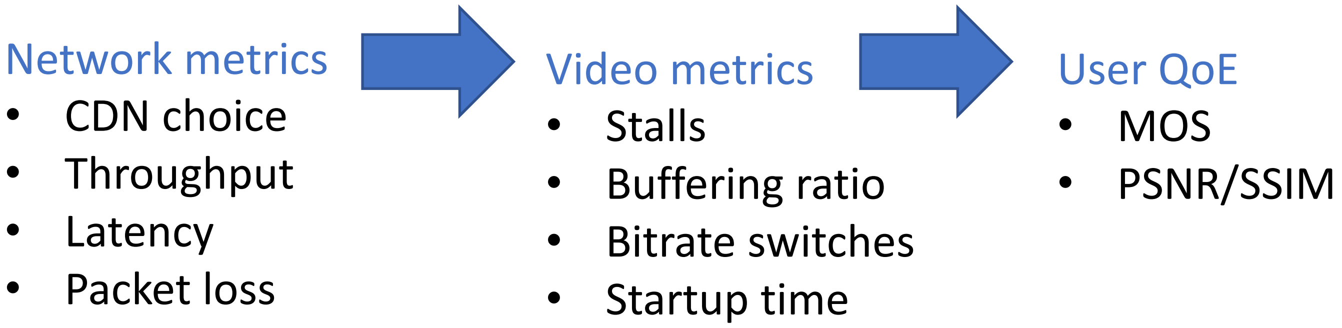
A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, H. Zhang

ACM Sigcomm 2013

Relationship between Metrics



	Engagement-centric	Actionable
PSNR-like (e.g., [17])	✗	✓
Opinion Scores(e.g., [6])	✓	✗
Network-level (e.g., bandwidth, latency [35])	✗	✓
Single metric (e.g., bitrate, buffering)	✗	✓
Naive learning	✗	✗
Our approach	✓	✓



Network metrics

- CDN choice
- Throughput
- Latency
- Packet loss

Video metrics

- Stalls
- Buffering ratio
- Bitrate switches
- Startup time

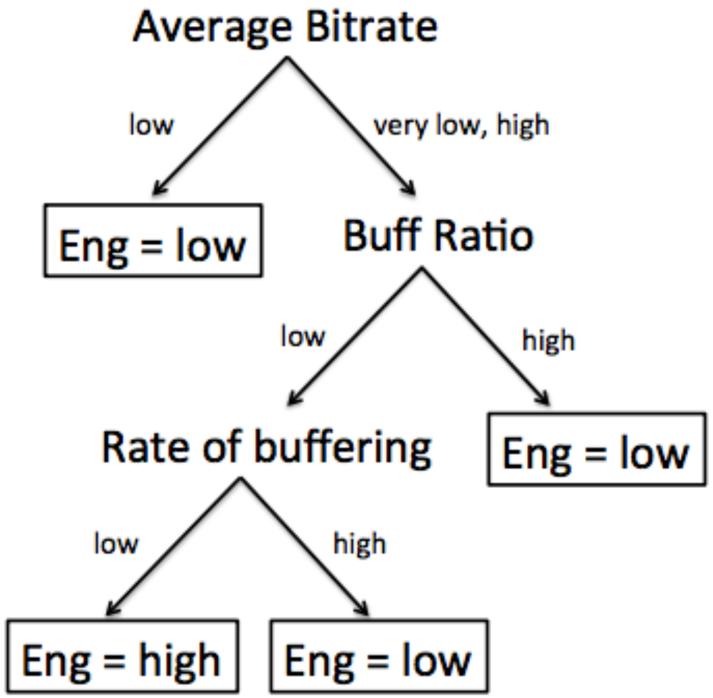
User QoE

- MOS
- PSNR/SSIM

Method

- Data from Conviva, a video delivery platform
  - 40 million sessions over 3 months in the US
  - VoD and live sports
  - Metrics collected by client

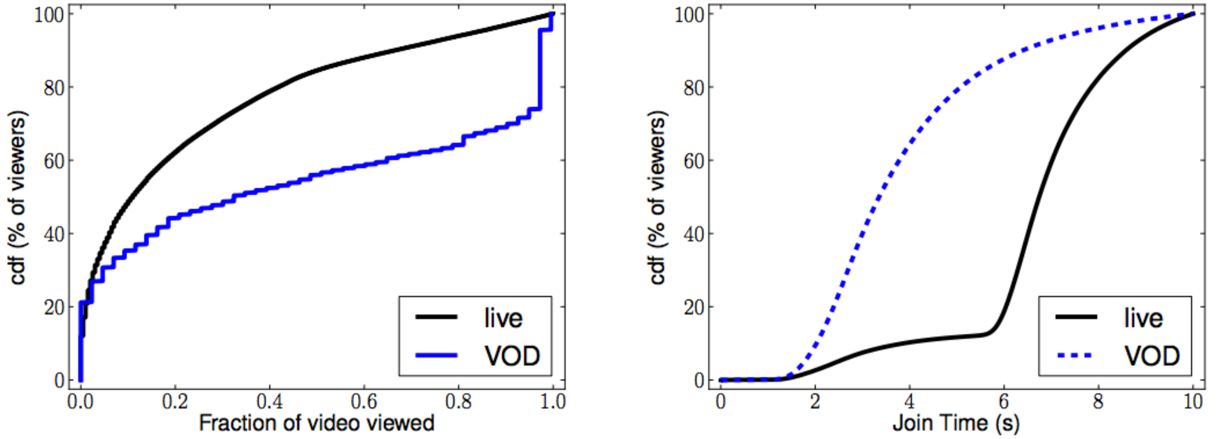
- Decision trees
  - Input: Video metrics
  - Output: Engagement metric
  - Bin these metrics



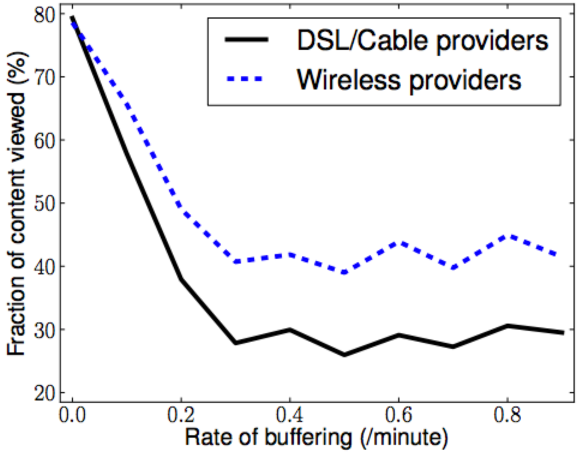
Live video

Confounding Factors?

- Type of video
  - Live
  - Video-on-demand
- User attributes
  - Location
  - Device (smartphones, tablets, laptop)
  - Connectivity (wireless, Ethernet)
- Temporal attributes
  - Time of day/week
  - Freshness



(a) User viewing pattern for live and VOD *(b)* Join time distribution for live and VOD



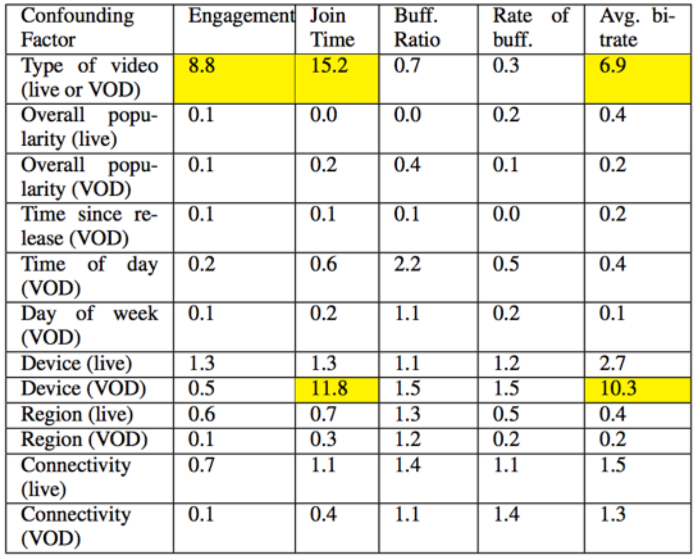
Detecting Confounding Factors

- Information gain metric
  - Entropy $H(Y) = -\Sigma_i P(Y=y_i) \log(P(Y=y_i))$
  - Conditional entropy $H(Y|X) = \Sigma_i P(X=x_i) H(Y|X=x_i)$
  - Information gain $H(Y) – H(Y|X)$

Y: the factor we are considering
X: the factor we could split along

- Determine which confounding factors have max information gain

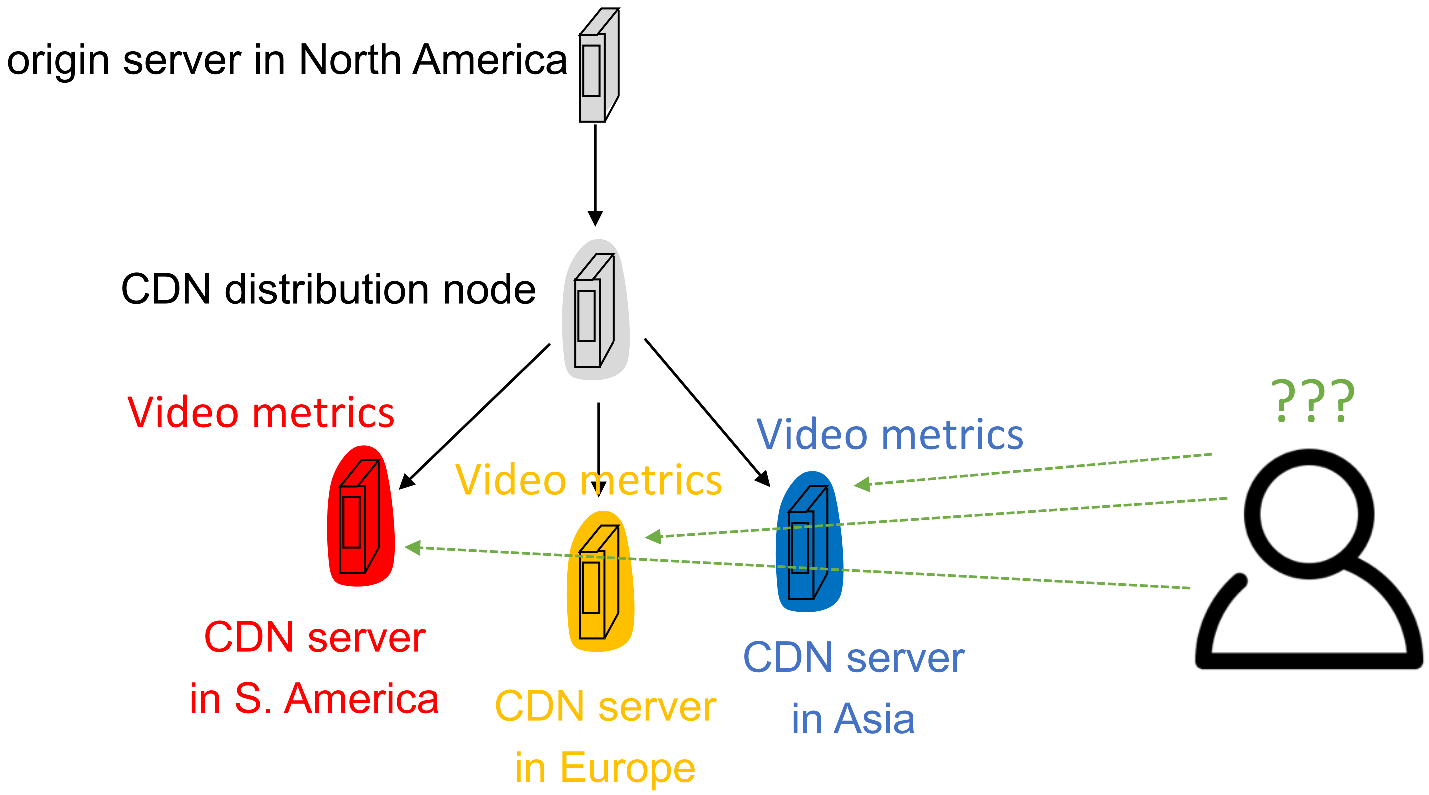
- Create a new decision tree for each confounding factor



Confounding Factor	Engagement	Join Time	Buff. Ratio	Rate of buff.	Avg. bitrate
Type of video (live or VOD)	8.8	15.2	0.7	0.3	6.9
Overall popularity (live)	0.1	0.0	0.0	0.2	0.4
Overall popularity (VOD)	0.1	0.2	0.4	0.1	0.2
Time since release (VOD)	0.1	0.1	0.1	0.0	0.2
Time of day (VOD)	0.2	0.6	2.2	0.5	0.4
Day of week (VOD)	0.1	0.2	1.1	0.2	0.1
Device (live)	1.3	1.3	1.1	1.2	2.7
Device (VOD)	0.5	11.8	1.5	1.5	10.3
Region (live)	0.6	0.7	1.3	0.5	0.4
Region (VOD)	0.1	0.3	1.2	0.2	0.2
Connectivity (live)	0.7	1.1	1.4	1.1	1.5
Connectivity (VOD)	0.1	0.4	1.1	1.4	1.3

Using the Model

- Output a decision tree that can predict the user QoE
- Use this to select CDN server



origin server in North America

CDN distribution node

Video metrics

Video metrics

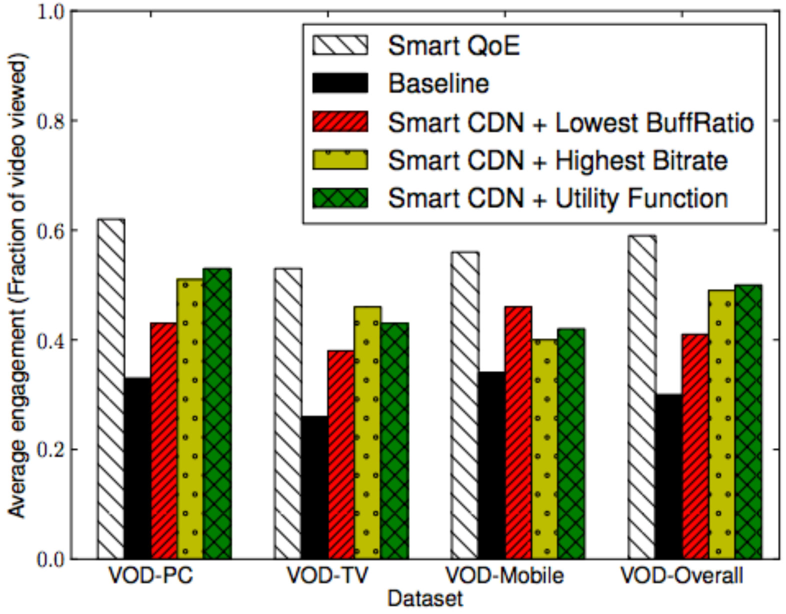
Video metrics

???

CDN server in S. America

CDN server in Europe

CDN server in Asia


Average engagement (Fraction of video viewed)
Smart QoE
Baseline
Smart CDN + Lowest BuffRatio
Smart CDN + Highest Bitrate
Smart CDN + Utility Function
VOD-PC
VOD-TV
VOD-Mobile
VOD-Overall
Dataset