

University of Modena and Reggio Emilia

Laboratorio di Comunicazioni Multimediali

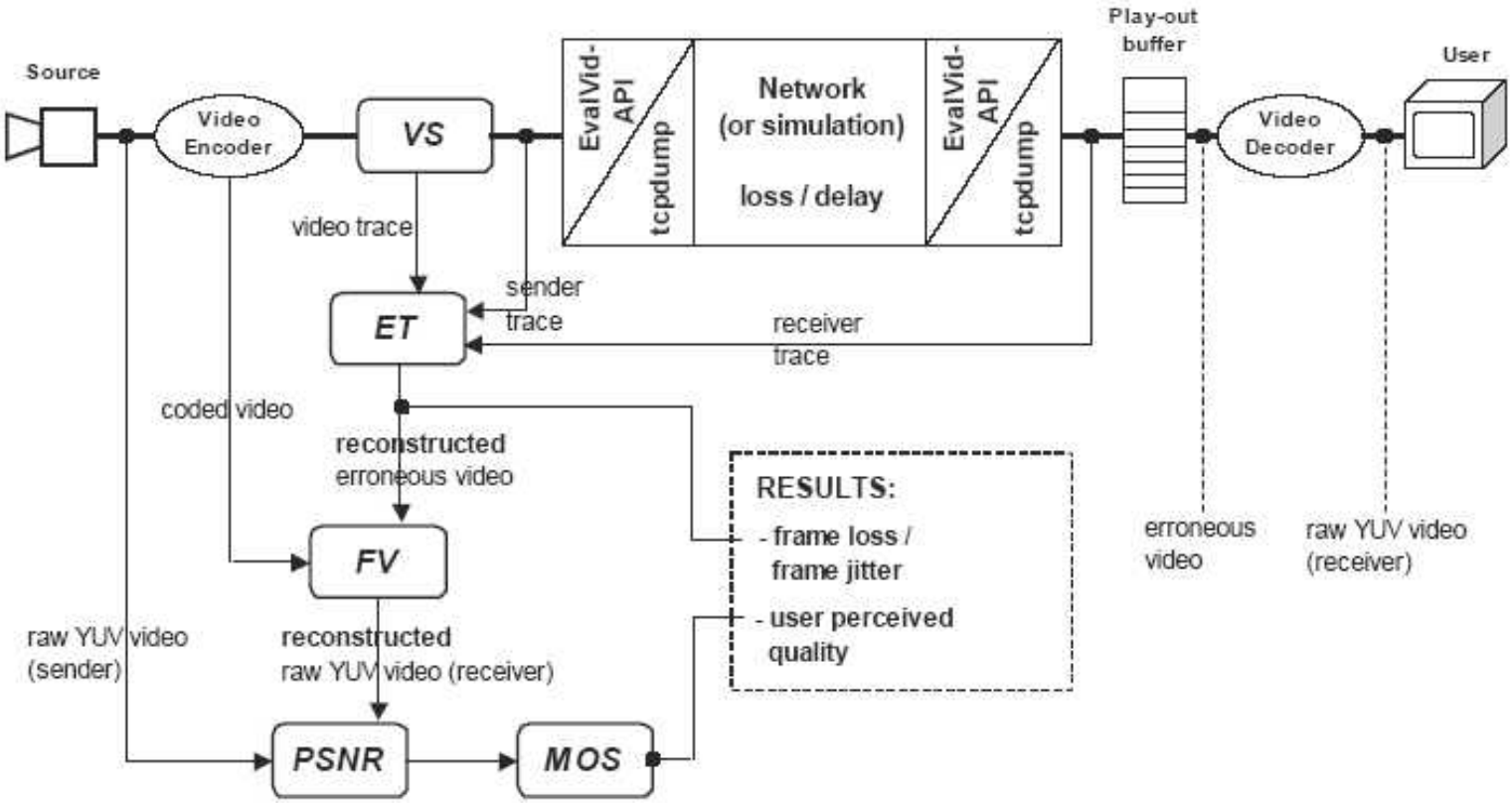
EVALVID

Daniela Saladino
(daniela.saladino@unimore.it)

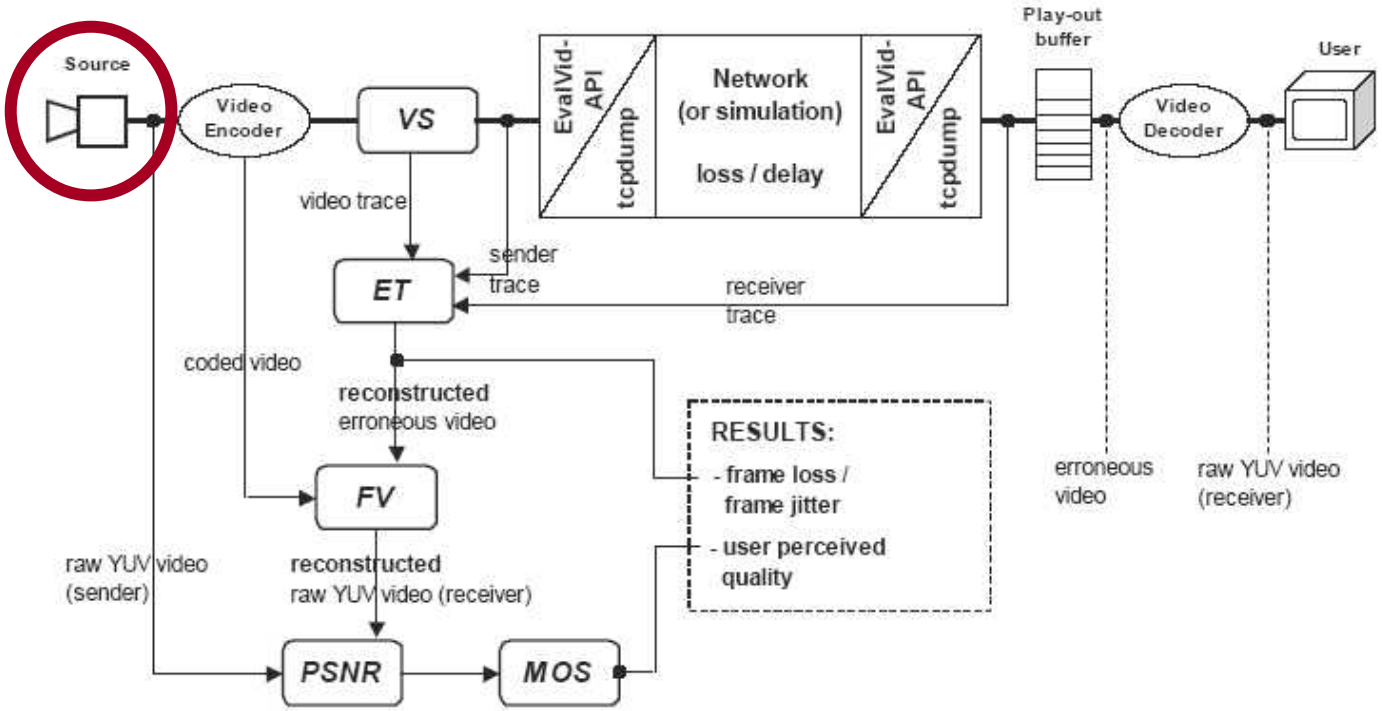
Features and Target

- Complete framework and tool-set for video quality evaluation:
  - packet/frame loss rate
  - delays
  - packet/frame jitter
  - PSNR and MOS metrics
- Modular structure

Overview of Evalvid

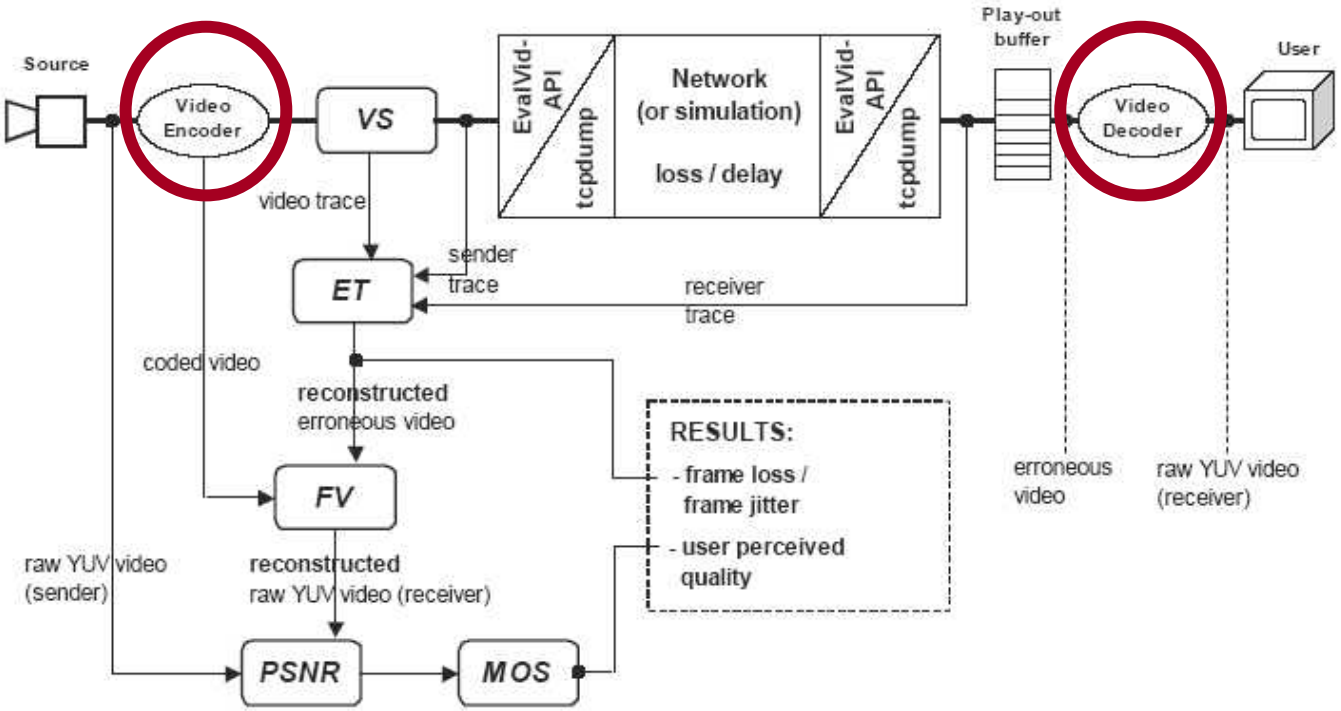


Components



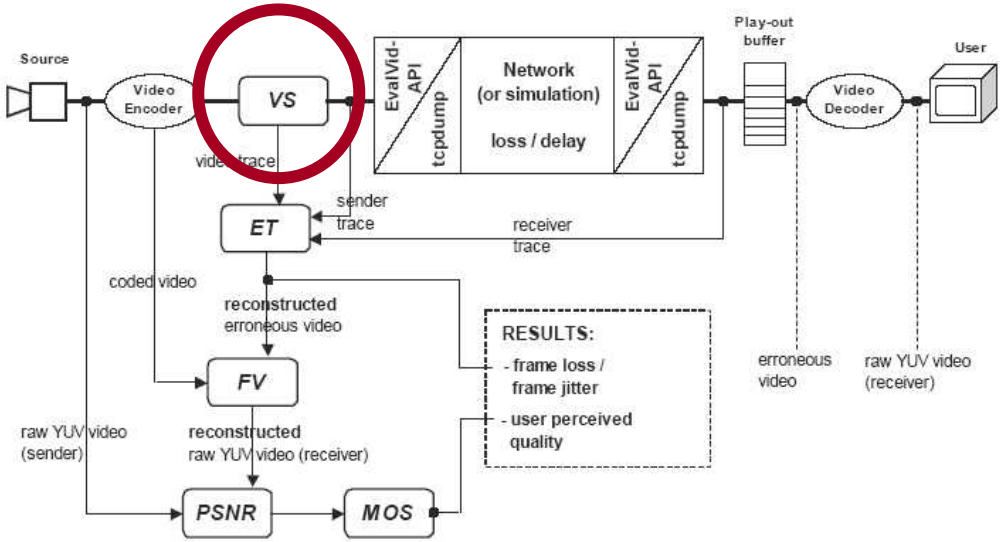
- ***Source***: raw (uncoded) video files are usually stored in YUV format

Components



- ***Video Encoder*** and ***Video Decoder***

Components



- ***VS (Video Sender)***: generates two trace files necessary for the subsequent video quality evaluation
  - ➔ sender trace file
  - ➔ video trace file

Examples

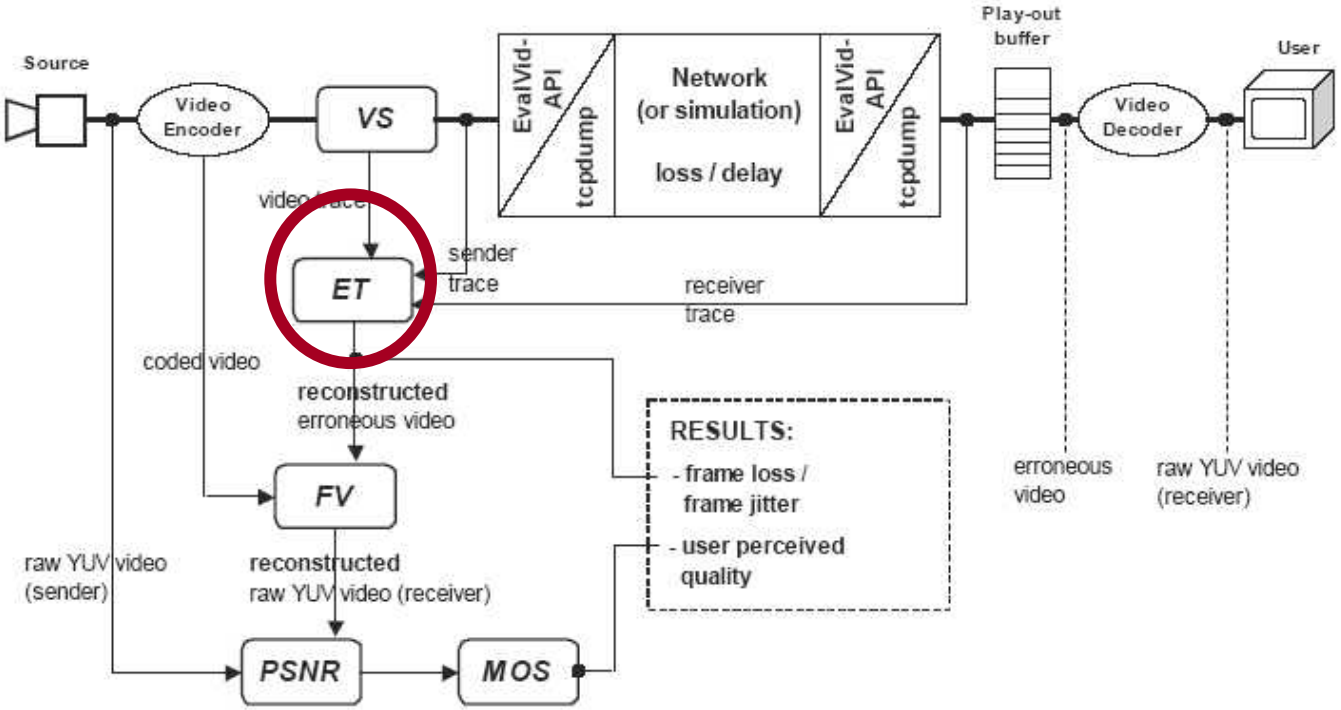
Frame Number	Frame Type	Frame Size	Number of UDP-packets	Sender Time
0	H	29	1 segment at	33 ms
1	I	3036	4 segments at	67 ms
2	P	659	1 segment at	99 ms
3	B	357	1 segment at	132 ms
4	B	374	1 segment at	165 ms
...	...	...	...	...

Video Trace File

Time stamp (sec)	Packet ID	Packet Type	Payload Size (bytes)
0.033333	0	udp	29
0.066666	1	udp	1000
0.066666	2	udp	1000
0.066666	3	udp	1000
0.066666	4	udp	36
0.099999	5	udp	659
0.133332	6	udp	357
0.166665	7	udp	374
...	...	...	...

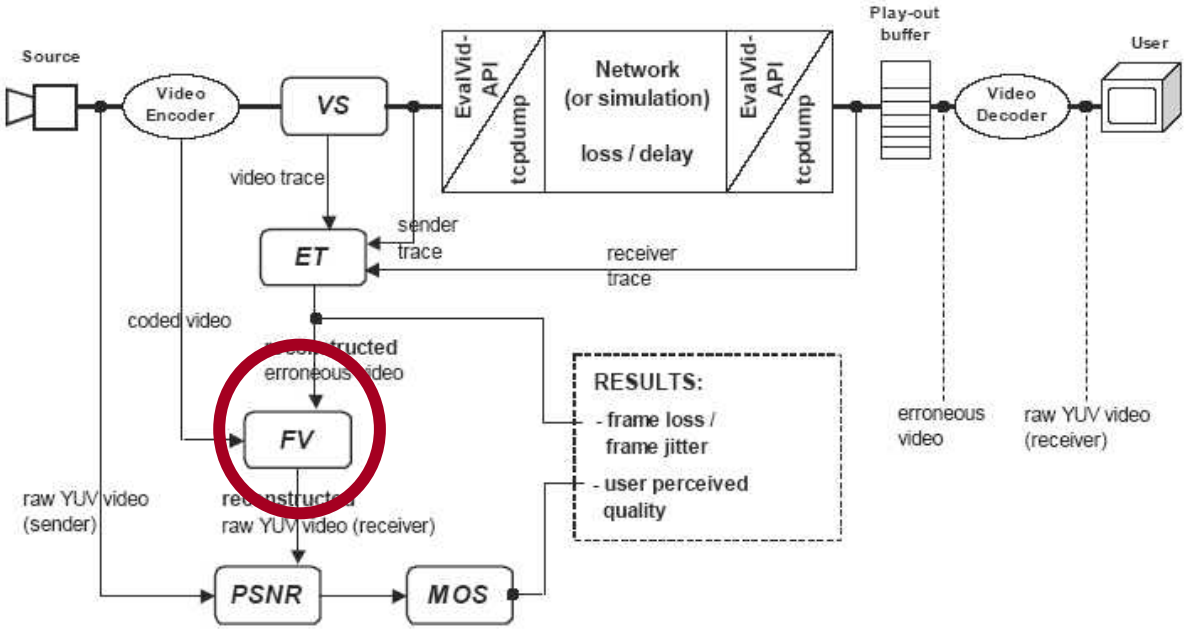
Sender Trace File

Components



- ***ET (Evaluate Trace)***: generates reconstructed erroneous video using also the receiver trace

Components



- ***FV (Fix Video)*:** is only needed if the used codec cannot provide lost frames (“empty” or the last decoded frames for lost frames)

PSNR (1/3)

- PSNR (Peak Signal to Noise Ratio) is an objective quality measure.
- It is a derivative of the well-known signal to noise ratio (SNR), which compares the signal energy to the error one.
- PSNR is usually expressed in terms of the logarithmic decibel scale.



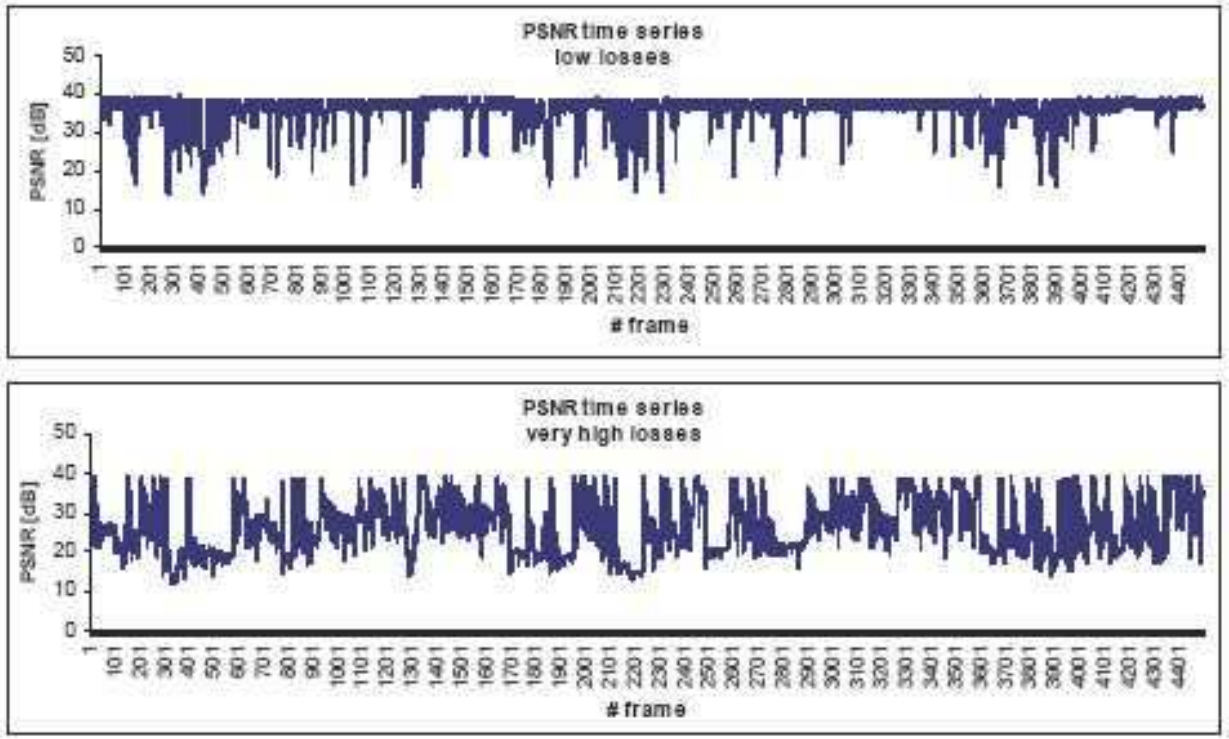
PSNR (2/3)

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

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

- where
  - MAX_I is the maximum possible pixel value of the image
  - *MSE* is the Mean Square Error:
    - I is the original image and K is the compressed one
    - MxN is the dimension of both images

PSNR (3/3)


PSNR time series
low losses
PSNR [dB]
50
40
30
20
10
0
frame
PSNR time series
very high losses
PSNR [dB]
50
40
30
20
10
0
frame

MOS (1/2)

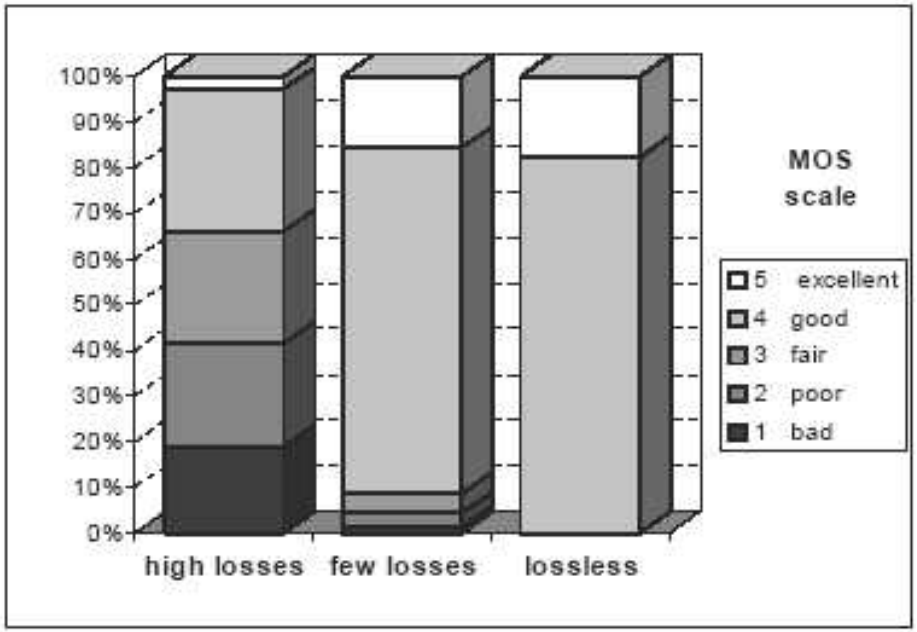
- MOS (Mean Opinion Score) is a subjective quality measure.
- MOS ranges from 1 (worst) and 5 (best):

Scale	Quality	Impairment
5	Excellent	Imperceptible
4	Good	Perceptible, but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

- PSNR approximated to the MOS scale:

PSNR [dB]	MOS
> 37	5 (Excellent)
31 - 37	4 (Good)
25 - 31	3 (Fair)
20 - 25	2 (Poor)
< 20	1 (Bad)

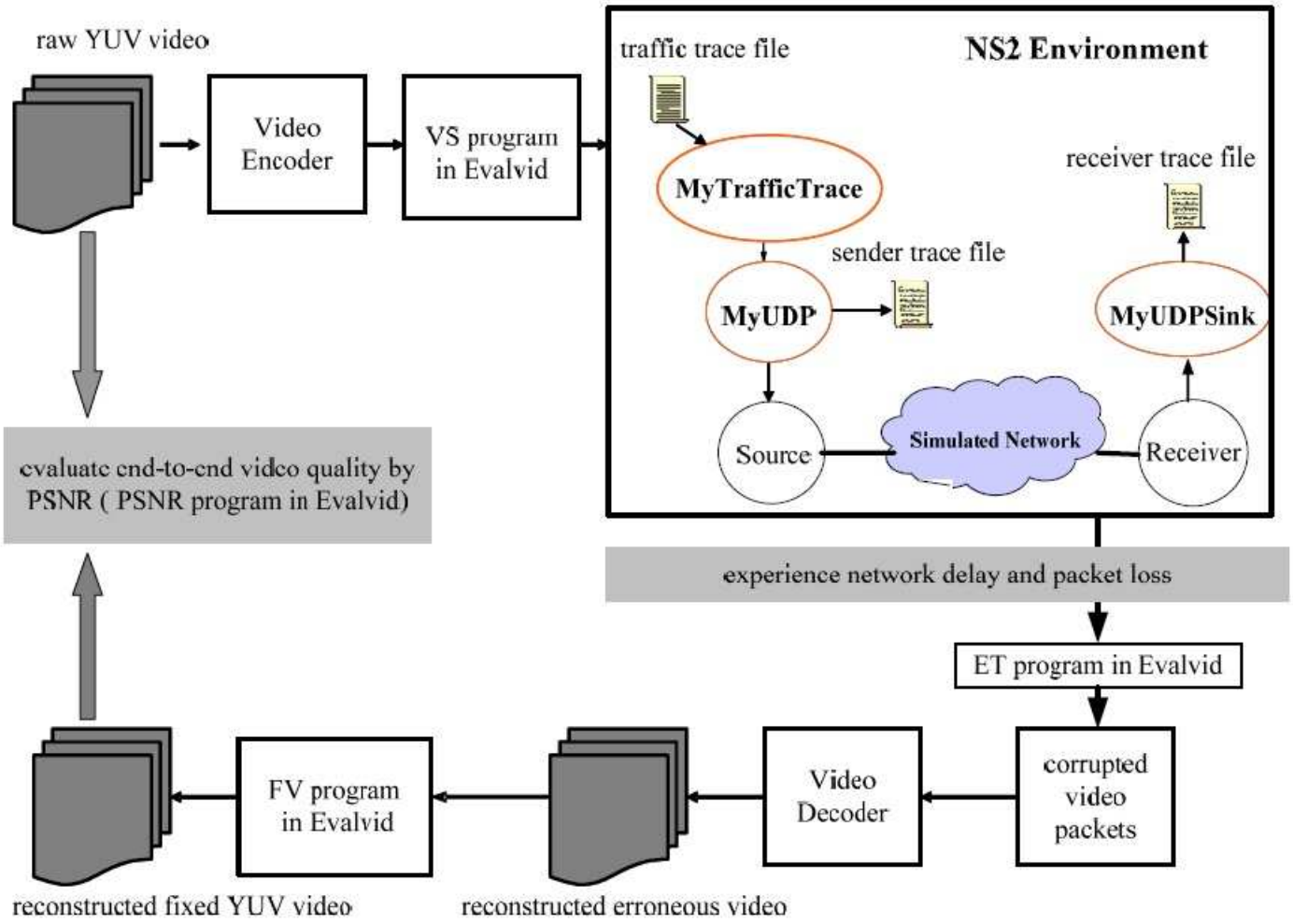
MOS (2/2)


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
high losses
few losses
lossless
MOS scale
5 excellent
4 good
3 fair
2 poor
1 bad

Evalvid and NS2

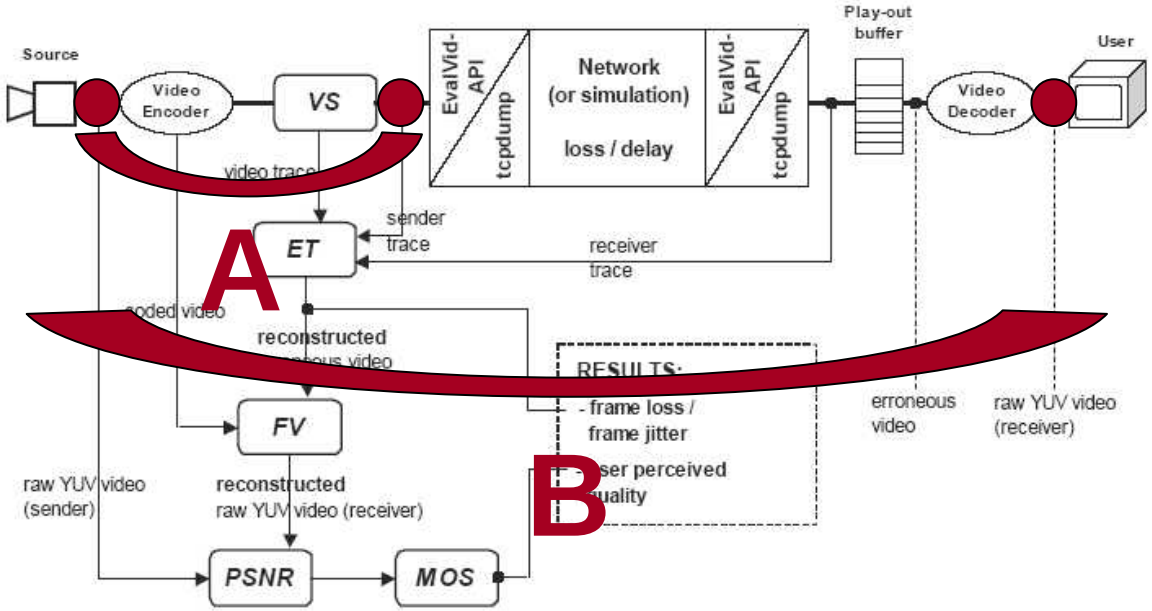
- To experiment video transmission using the NS2 simulator
- Simulation produces a receiver trace file containing information necessary to reconstruct the possibly corrupted video at the receiver side.

Evalvid and NS2



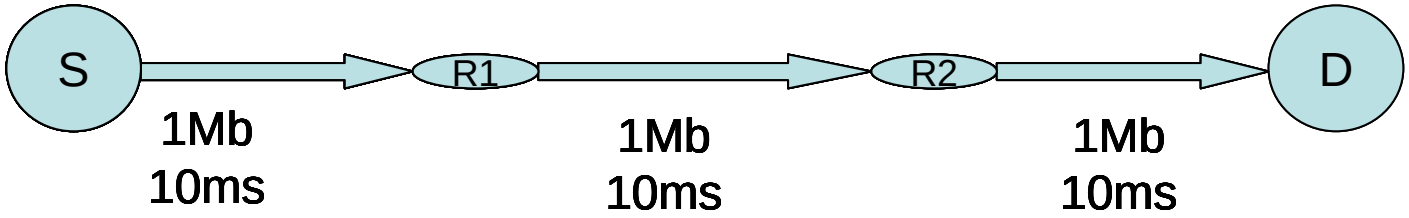
Practise

- A video coded employing MPEG-4
- Comparison between PSNR *before* video transmission (A) and PSNR *after* video transmission (B) using a network simulated by **NS2** with and without transmission errors

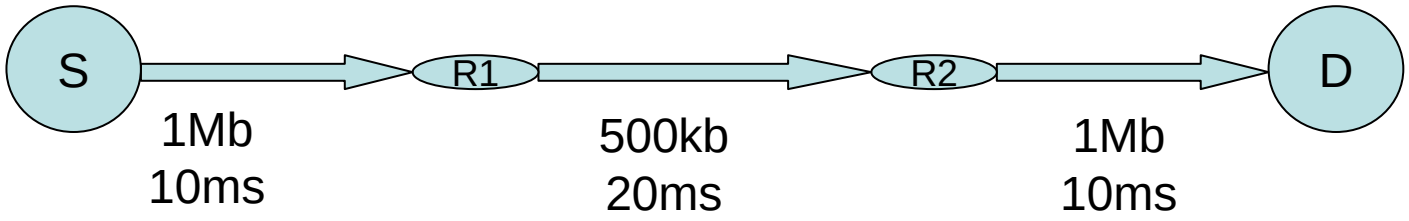


Evalvid instructions and NS2 (1/4)

- Run VirtualBox
- Import virtual machine in VirtualBox from /opt/src/evalvid
- “evalvid” folder: exercise without transmission errors



- “evalvidConErrore” folder: exercise with transmission errors





Evalvid instructions and NS2 (2/4)

- To decode the video files obtaining a YUV (raw) video:
  - ffmpeg -i video.264 video_raw.yuv
- To create a compressed raw video: MPEG-4
  - ffmpeg -s cif -r 30 -b 64000 -bt 3200 -g 30 -i video_raw.yuv –vcodec mpeg4 video_cod.m4v
- To create a MP4 files containing the video samples (frames) and a hint track which describes how to packetize the frames for the transport with RTP:
  - ./mp4box -hint -mtu 1024 -fps 30 -add video_cod.m4v video_encaps.mp4

Evalvid instructions and NS2 (3/4)

- To obtain the YUV file created by decoding the coded video
  - ffmpeg -i video_encaps.mp4 video_ref_raw.yuv
- To compute PSNR that shows the codec impact on video quality
  - ./psnr 352 288 420 video_raw.yuv video_ref_raw.yuv > psnr_prima.txt
- To send a hinted mp4-file per RTP/UDP to a specified destination host
  - ./mp4trace -f -s 192.168.0.2 12346 video_encaps.mp4 > st_video
  - now you have the **video trace** => st_video

Evalvid instructions and NS2 (3/4)

- To simulate a real network execute ns2 script
  - ns rete.tcl
  - now you have also a **sender trace** => sd_video (sender time of each packet) and a **receiver trace** => rd_video (received time of each packet)
- To reconstruct the transmitted video as it is seen by the receiver
  - ./etmp4 –f –x sd_video rd_video st_video video_encaps.mp4 video_reconstr
  - this generates a (possibly corrupted) video file
- To decode the received video to YUV (raw) format
  - ffmpeg -i video_reconstr.mp4 video_reconstr_raw.yuv

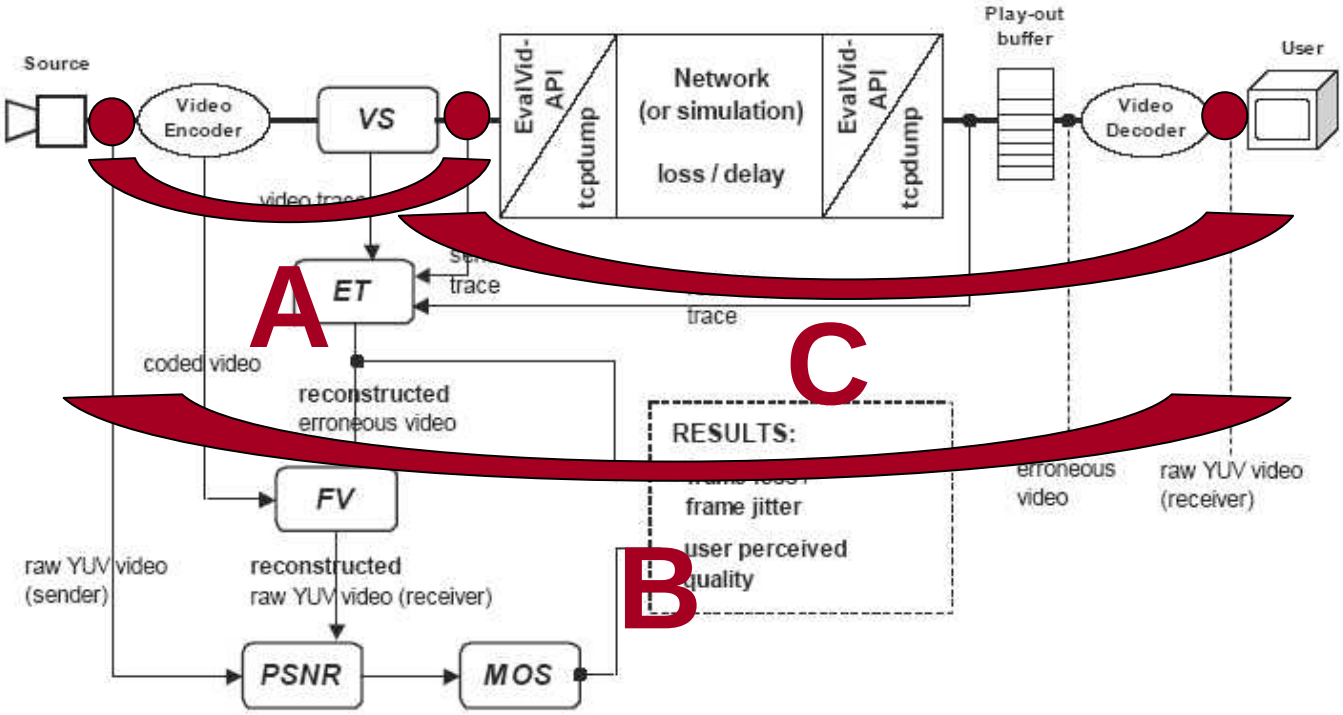


Evalvid instructions and NS2 (4/4)

- To compute the PSNR that shows the transmission impact on video quality
  - ./psnr 352 288 420 video_ref_raw.yuv video_reconstr_raw.yuv > psnr_dopo.txt
- To create graphics
  - ns grafico_psnr.tcl

Esercizio (1/2)

- Analizzare un altro video codificato sia in MPEG-4 che in H.264 con diverse caratteristiche della rete (sempre con e senza perdita)
- Calcolare il PSNR e MOS per gli scenari mostrati in figura e graficarli



Esercizio (2/2)

- Graficare inoltre
  - la frame loss (%)
    - percentuale di frame I, B e P persi
    - percentuale di frame complessivamente persi
  - il ritardo end-to-end dei frame (PDF e CDF)
- Riportare tutti i parametri del video analizzato (bitrate, numero di frame, larghezza e altezza dei frame, ecc.)
- Usare come riferimento:
  - http://www.tkn.tu-berlin.de/research/evalvid/EvalVid/example.html

- For more information:
  - http://www.tkn.tu-berlin.de/research/evalvid/
  - J. Klaue, B. Rathke, and A. Wolisz, "*EvalVid - A Framework for Video Transmission and Quality Evaluation*"
  - Chih-Heng Ke, Ce-Kuen Shieh, Wen-Shyang Hwang, Artur Ziviani, "*An Evaluation Framework for More Realistic Simulations of MPEG Video Transmission*"

For clarifications:

Daniela Saladino: daniela.saladino@unimore.it