Bit-Error Resilient Packetization for Streaming H.264/AVC Video

Jari Korhonen, and Pascal Frossard

MV'07, 28. September 2007

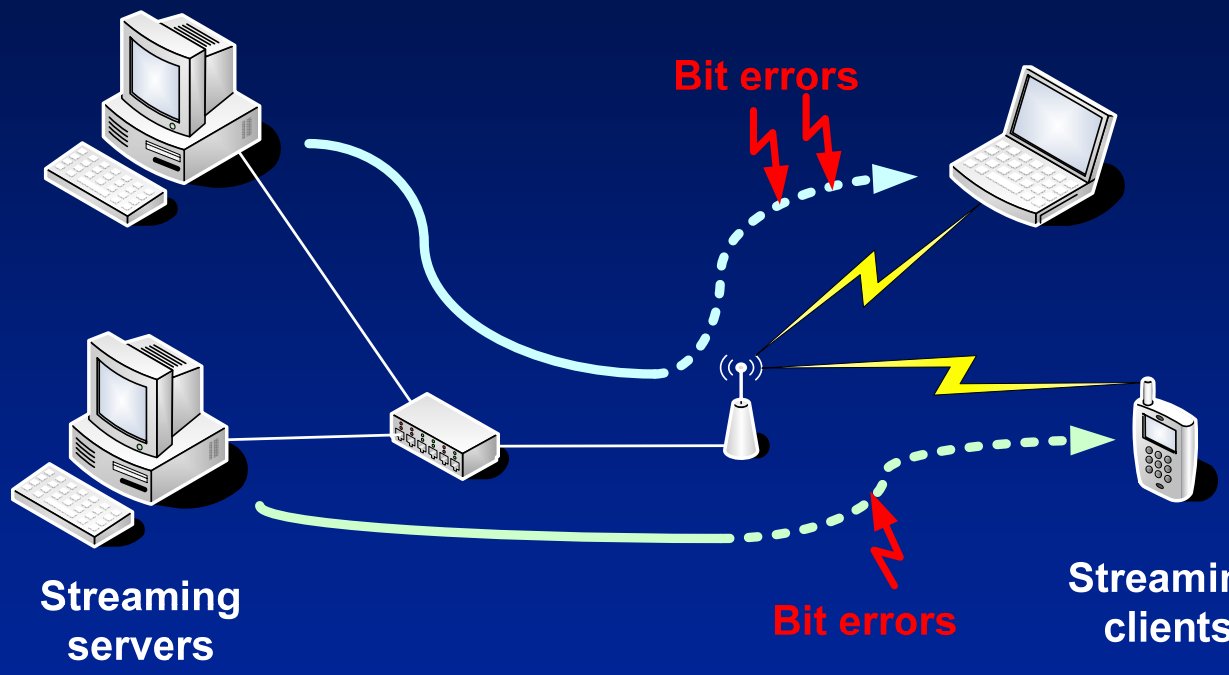
Signal Processing Institute
Swiss Federal Institute of Technology, Lausanne

Outline

- Motivation and introduction
- Error resilience in video streaming
  - *General issues*
  - *Proposed approach*
- Simulation setup
- Simulation results
- Discussion and conclusions

Motivation and introduction

- Bit errors in a radio channel may communication channel increase retransmission overhead and reduce the available capacity



- Problems with streaming applications especially!

Packet loss resilience

- Resilience to packet losses
  - *If a video frame or a slice of a video frame is lost, decoder can try to conceal errors*
    - Interpolation from adjacent slices/frames
  - *Small missing areas are usually easier to conceal than large areas*
    - Motivation to use small packets, but this leads to increased header overhead

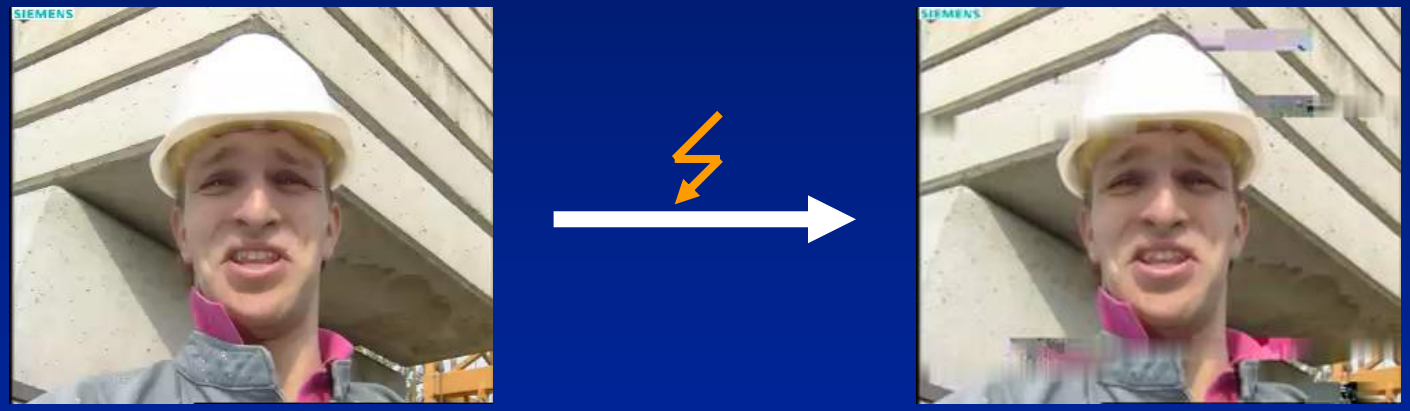
Original video stream

Slices lost in a lossy channel

After concealment

Bit error resilience

- Data corrupted by bit errors can be passed to the application instead of discarding or retransmitting
  - *Only the most critical sections are protected (UDP Lite)*
  - *Number of retransmissions can be reduced and channel capacity used more efficiently*
  - *Problem: can the application use the corrupted data?*
    - For example, H.264/AVC standard does not support bit error resilience



Original video stream

Erroneous data decoded

Bit errors in different types of frames

Erroneous chroma components

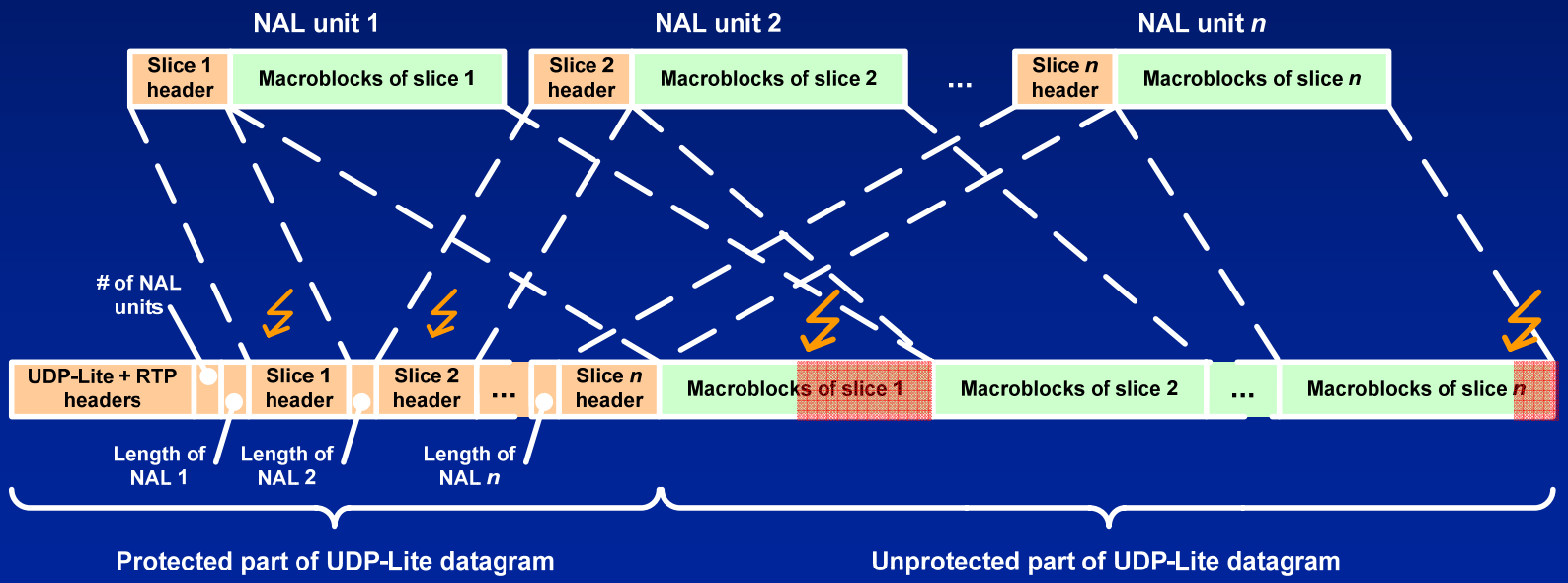
Erroneous motion vectors

Bit errors in an I-frame

Bit errors in a P-frame

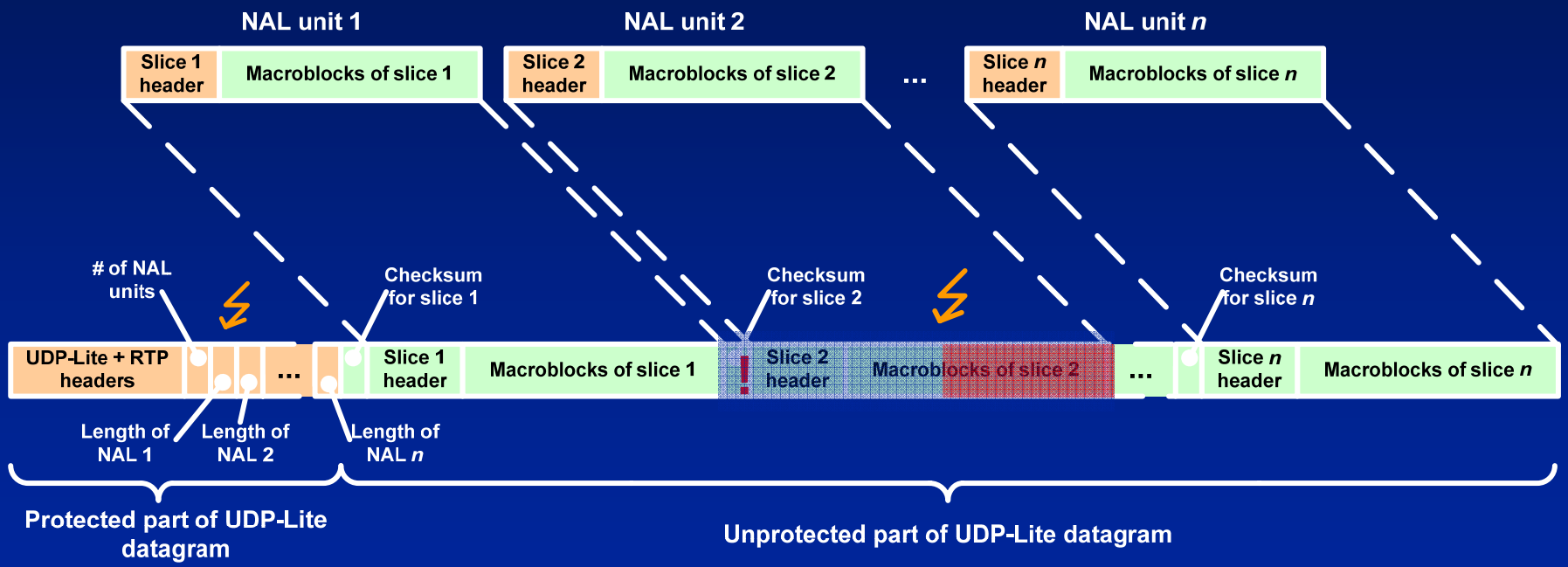
Proposed error resilient packetization: decode all NALUs (err)

- Small NALUs preferable, several NALUs per packet
- NALU headers reallocated and protected by UDP-Lite partial checksum
- Try to decode all NALUs in spite of possible errors!



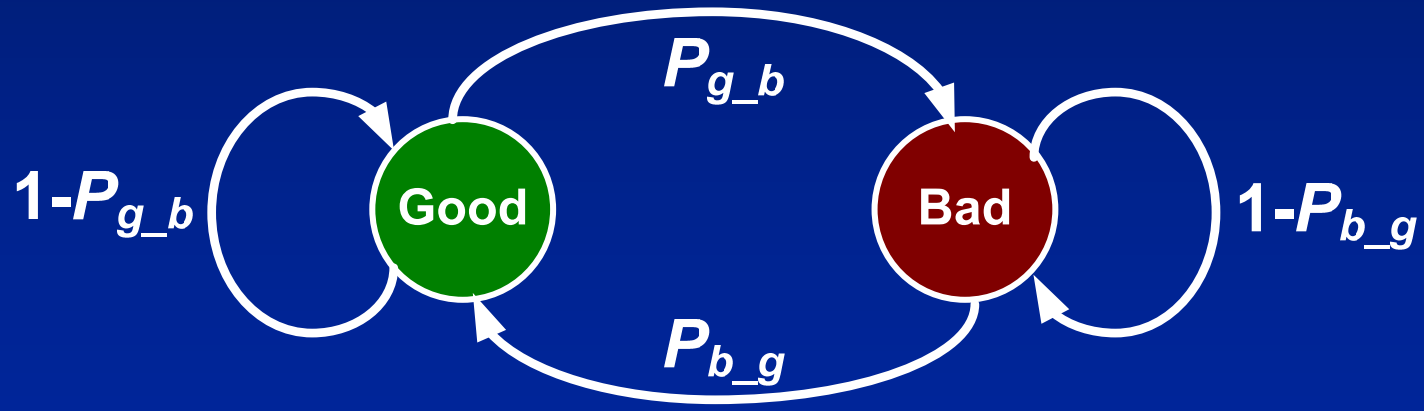
Proposed error resilient packetization: discard erroneous NALUs (loss)

- Another alternative: each NALU is protected by an individual checksum and damaged NALUs are discarded at the application layer



Simulation setup (1)

- Both two proposed packetization schemes (referred as ‘err’ and ‘loss’) were implemented and tested in an offline simulation
- Bit errors in a wireless radio channel simulated with a two-state model
  - *In the ‘Good’ state, all bits are correct*
  - *In the ‘Bad’ state, each bit is inverted at probability* $P_{corrupted}$
- It is assumed that bit errors in protected area are corrected by retransmissions



Simulation setup (2)

- Modified H.264/AVC reference codec (JM 11.0)
  - *Implemented simple bit error management*
- Three different maximum NALU sizes
  - *Long: 1350 bytes, ~1 NALU per packet*
  - *Medium: 450 bytes, ~3 NALUs per packet*
  - *Short: 150 bytes, ~10 NALUs per packet*
- Three different bit error patterns
  - *Bernoulli process ($P_{g\_b}=10^{-3}$, $P_{b\_g}=1$, $P_{corr}=1$)*
  - *Short bursts ($P_{g\_b}=10^{-4}$, $P_{b\_g}=0.05$, $P_{corr}=0.5$)*
  - *Long bursts ($P_{g\_b}=2\cdot10^{-5}$, $P_{b\_g}=0.015$, $P_{corr}=0.5$)*
  - *In each case, the proportion of corrupted bits is ~10^{-3}*
- Three test sequences
  - *Foreman CIF @ 256 kbit/s and @ 512 kbit/s*
  - *Soccer CIF @ 512 kbit/s*

Results: total PSNR values (Foreman @ 512 kbit/s)

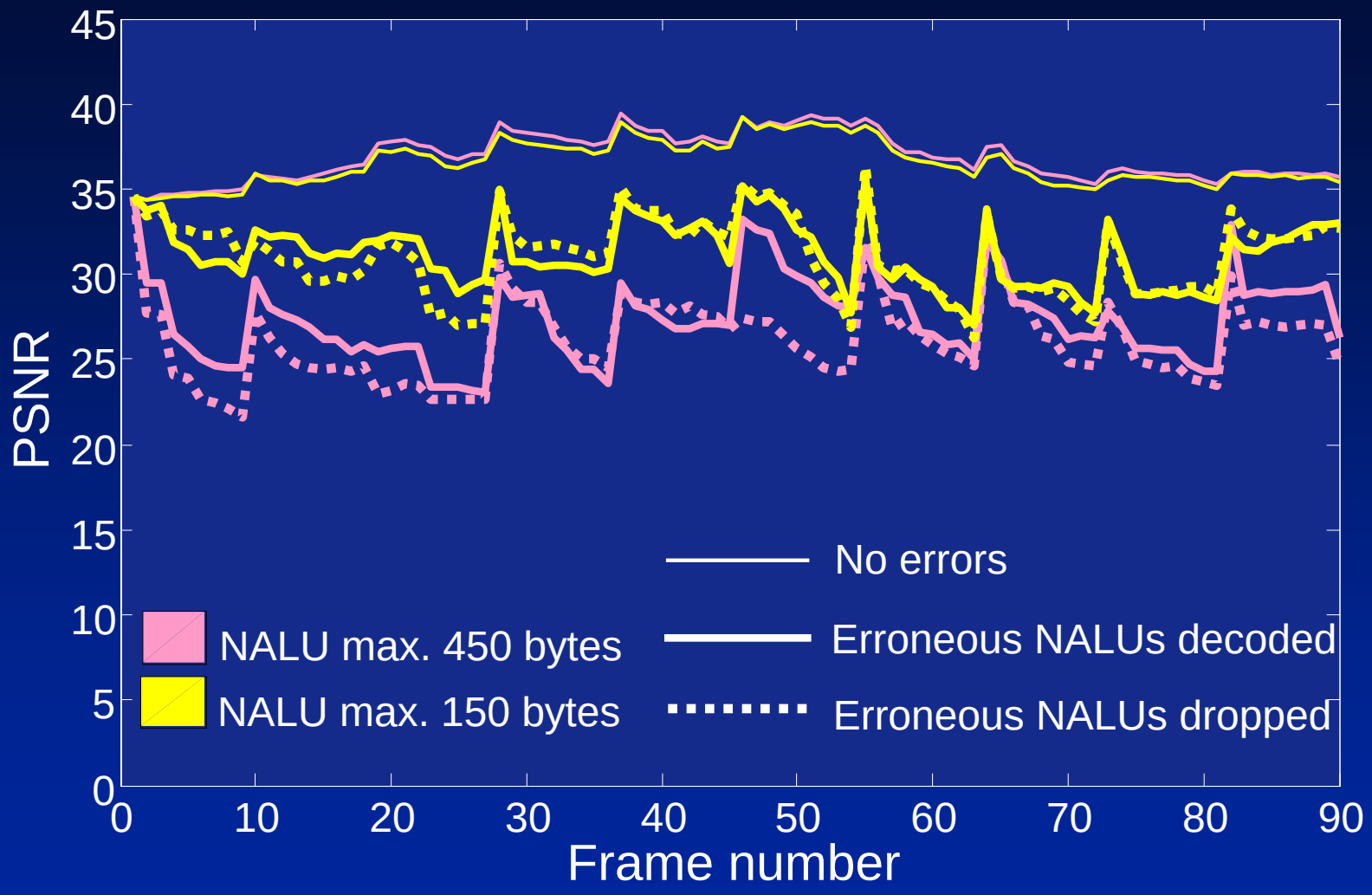
	No bit errors	Random errors	Short bursts	Long bursts
150-loss	38.78	22.83	31.72	**35.54**
150-err	38.78	**25.54**	**32.07**	35.25
450-loss	39.11	X	26.45	29.99
450-err	39.11	**21.82**	**29.00**	**32.33**
1350-loss (baseline)	39.32	X	X	27.34
1350-err	39.32	**18.74**	**25.66**	**29.03**

Qualitative difference: discarding vs. decoding

Erroneous slices discarded and concealed

Erroneous slices attempted to decode

Simulation results: PSNR frame by frame (short bursts)



Signal Processing Institute
Swiss Federal Institute of Technology, Lausanne

Conclusions

- In wireless links bit errors cause retransmission overhead and reduce the available link capacity
- One solution for video streaming is to pass corrupted packets up to the application
  - *Our proposal: pack several small NALUs in each packet, use error protection for the critical data only*
  - *Increased error resilience can be achieved by using small NALUs and decoding corrupted NALUs instead of discarding*
  - *Error pattern matters: a single bit error may be as harmful as a burst of bit errors!*

Thank You!

- Questions, comments?