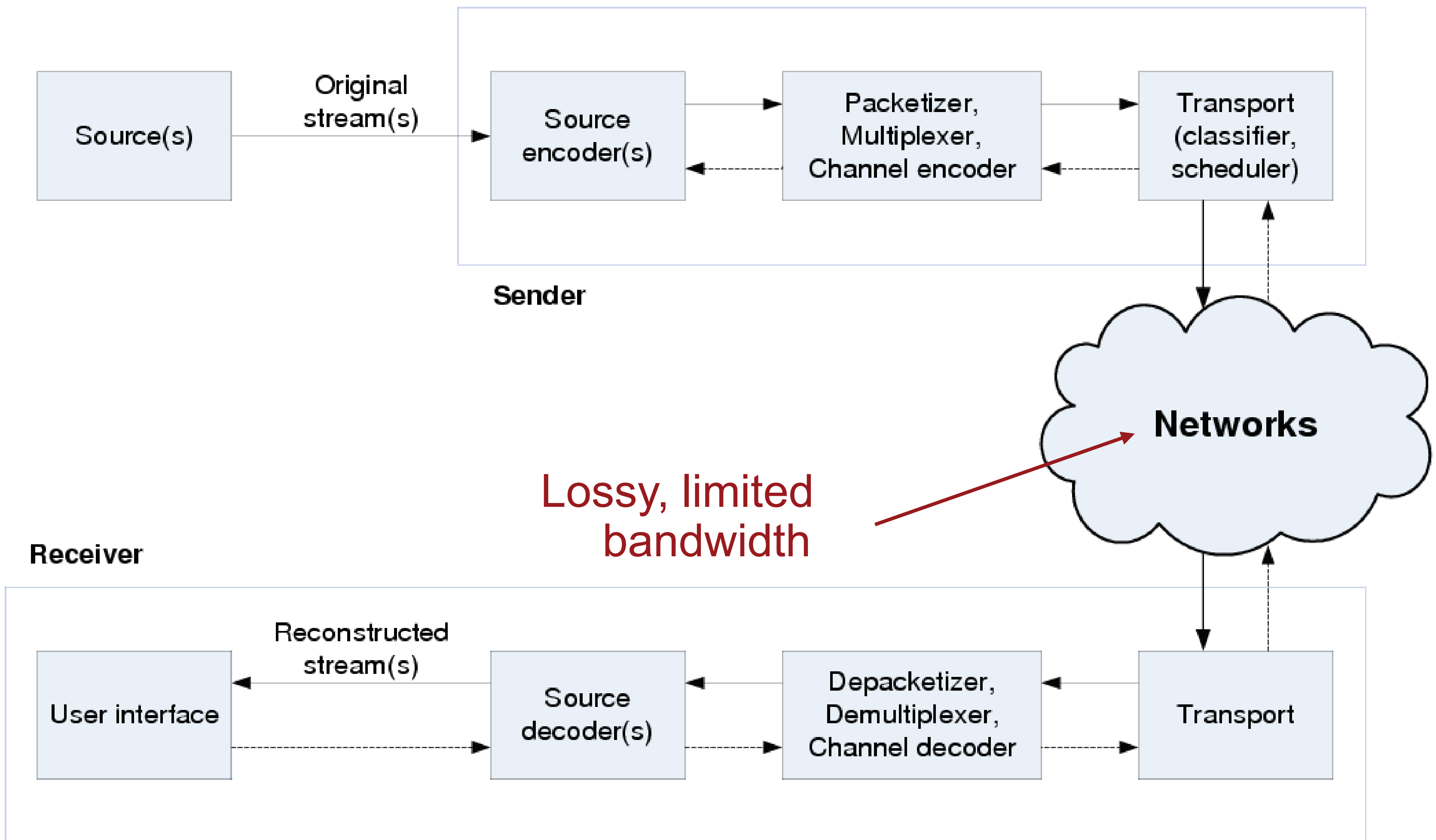
Packet prioritizing & delivering for multimedia streaming

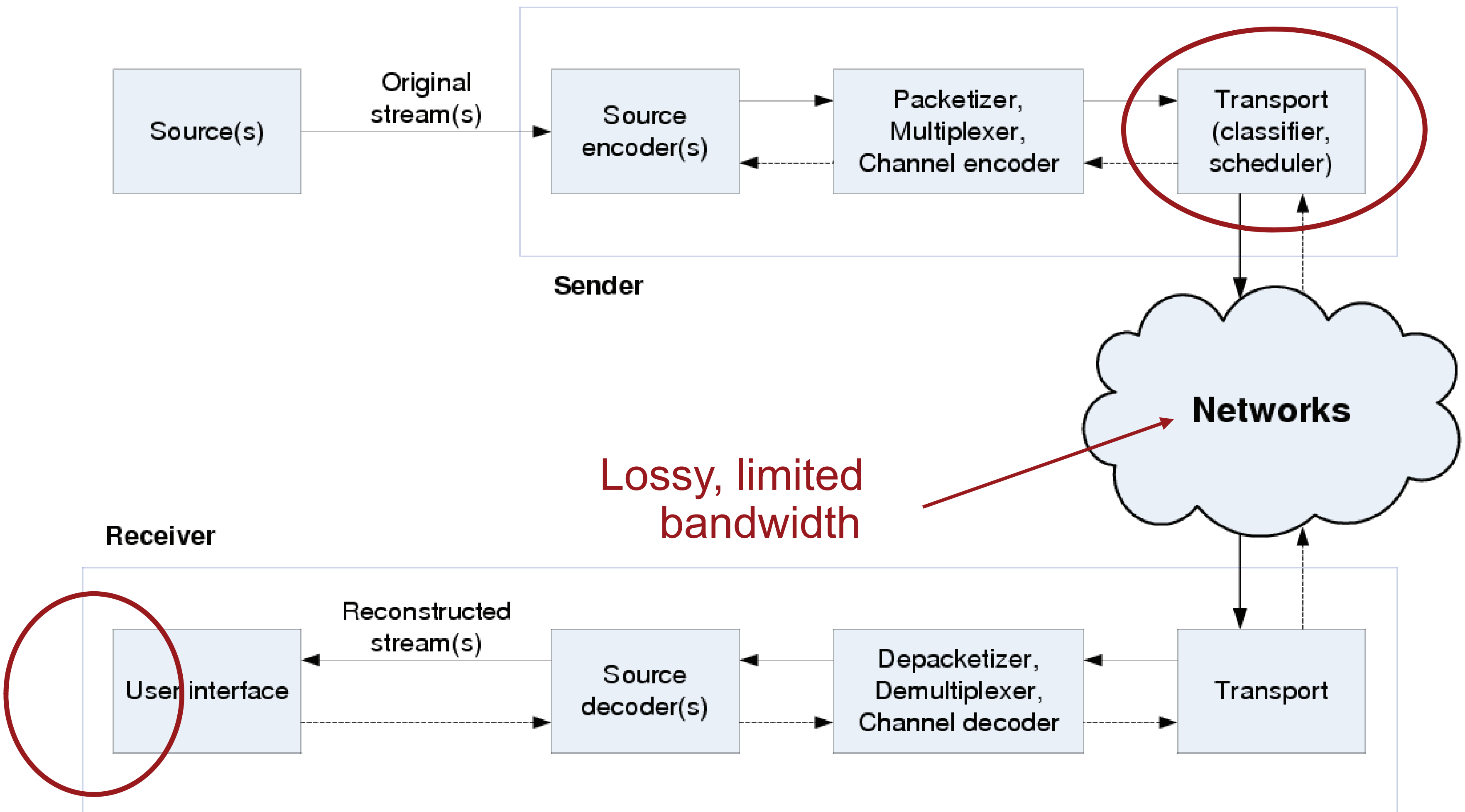
Candidate: **Vu-Thanh Nguyen**
Advisor: **Dr. Ee-Chien Chang**
Thesis committee: **Dr. Roger Zimmermann**
Dr. Wang Ye
Dr. Wei-Tsang Ooi

School of Computing, National University of Singapore
29 July 2008

A multimedia streaming system

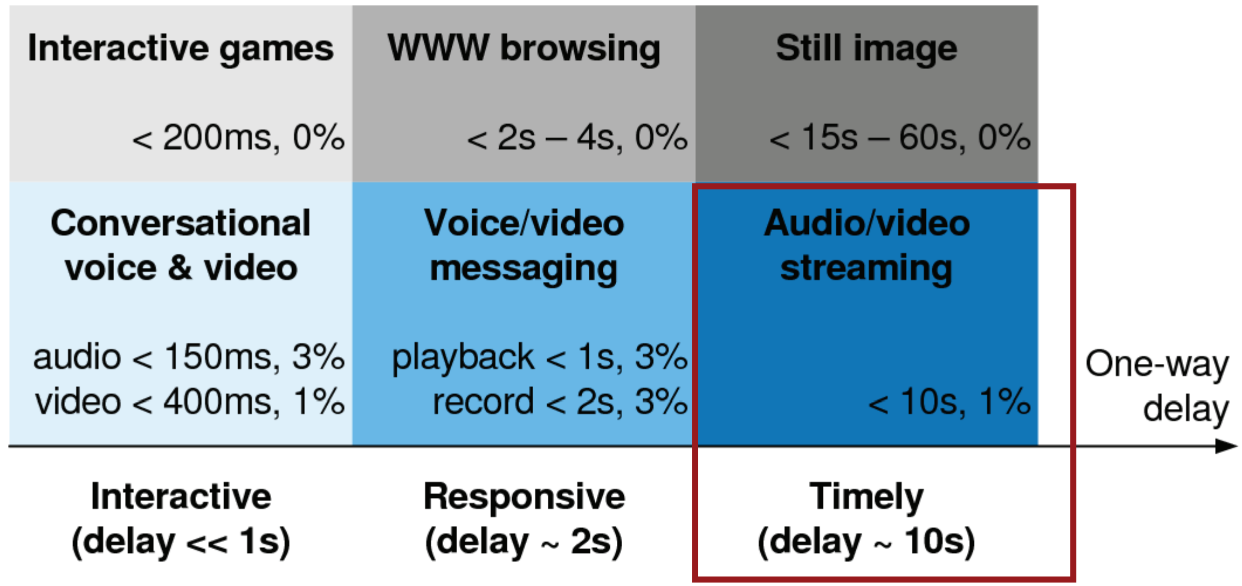


Our focus


Source(s)
Original stream(s)
Source encoder(s)
Packetizer, Multiplexer, Channel encoder
Transport (classifier, scheduler)
Sender
Networks
Lossy, limited bandwidth
Receiver
User interface
Reconstructed stream(s)
Source decoder(s)
Depacketizer, Demultiplexer, Channel decoder
Transport

Users & network

- User requirements (ITU G.1010)



- Network
  - Packet loss ratio
    - 1999–2001: < 1% (most links), 1-10% (15% links), 10% (< 1% links)
    - 2006–2008: 5–12% (Internet Traffic Report)
  - RTT: *3G*: 100ms, *Internet*: 134–160ms, *GPRS*: 600–1000ms

- Retransmission is possible*
  - Link-layer retransmission: if 1s delay is allowed
  - Application-layer retransmission: if 2-3s is acceptable

* Minoru Etoh & Takeshi Yoshimura. Advances in wireless video delivery. *IEEE Proc.*, 2005

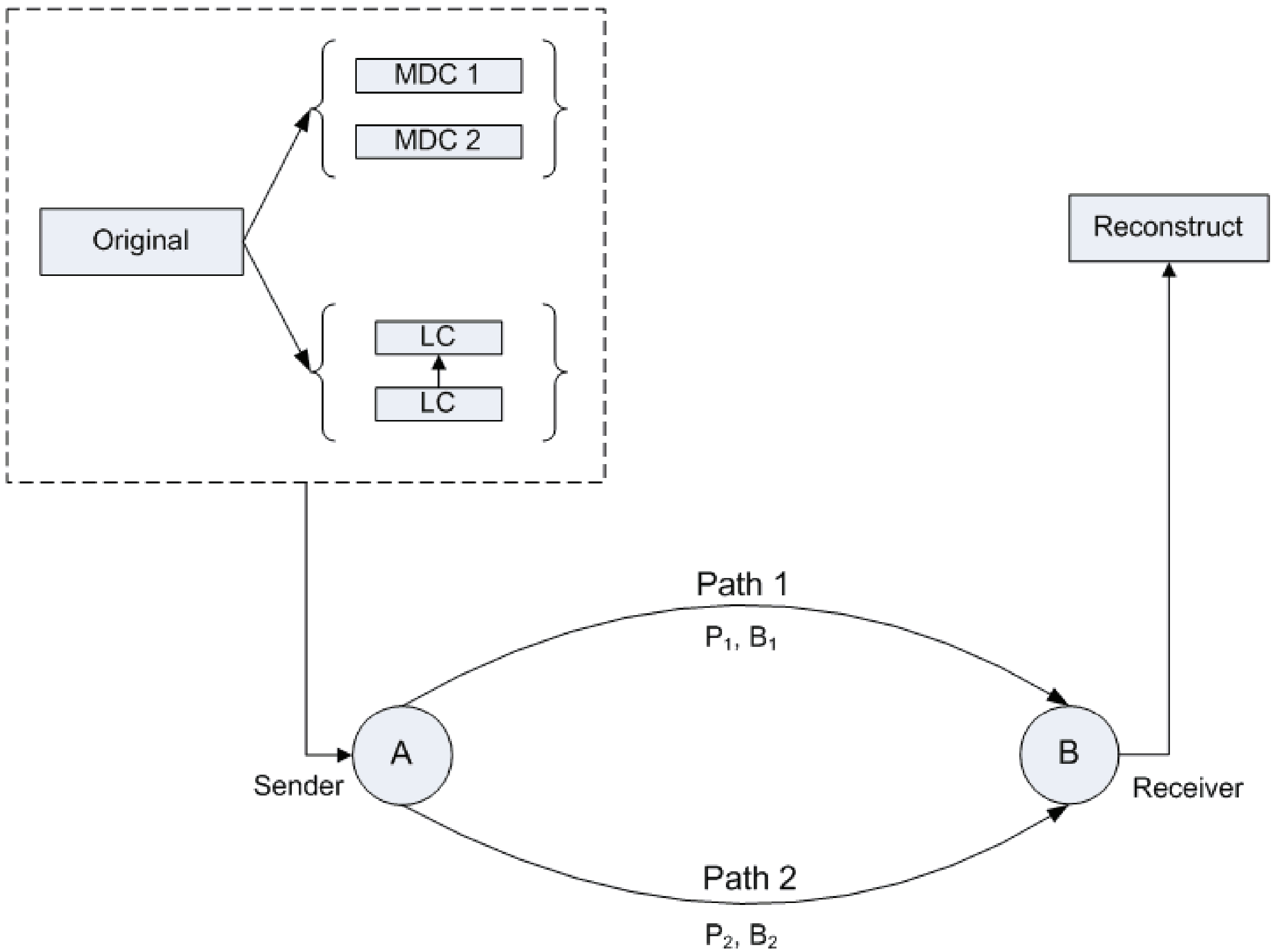
Motivation

- Packet delivery
  - User requirements aren’t that strict (10s delay for streaming)
  - For streaming prioritized packets
    - QoS architecture is not a prerequisite
    - Sender can use UEP, retransmission
- Packet prioritization
  - Current systems: ***syntax data*** is important
  - For users: ***semantic content*** is important
  - Should be based on users’ interests (semantic content), rather than systems’ focus (syntax data)

Roadmap

- Motivation
- The questions, our approaches & results
  - Should we ***prioritize*** packets?
  - ***What and how*** to prioritize packets?
  - How to ***schedule*** prioritized packets?
- Conclusion

Q1. Should we ***prioritize*** packets?



Q1. Related works

- Packet distribution
  - Apostolopoulos (2001): two independent streams (even & odd) over two paths
  - Liang & Girod (2001): similar system for voice with MDC ⇨ path to send next packet = path received last ACK
  - Chakareski & Girod (2003): minimize distortion based on ACK feedback

- Common belief: LC is worse than MDC
  - when the application requires short delay but network has long RTT or no feedback channel
  - at high packet loss rate

Q1. Our observations

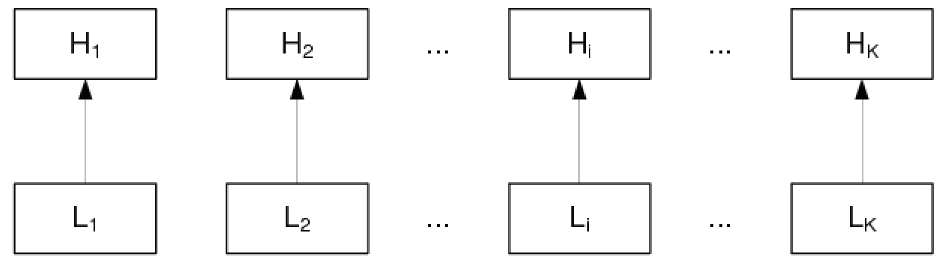
- MDC: LC + redundancies
  - LC is more bandwidth efficient than MDC
- Packet transmission
  - Better quality can be achieved with better allocation algorithms
    - For MDC: packets are equally important ⇨ can send any packet along any path
    - For LC: packets have different priorities ⇨ can decide which packet to send over which path
  - Packet ACK is not a prerequisite

Q1. Optimization framework

- Input
  - a set of m disjoint paths $P = \{P_1, P_2, \ldots, P_m\}$, their probability of loss p_i
  - the dependencies between packets, and the rate-distortion function.
- Constraint: effective bandwidths B_i of paths
- Output: allocation - which packet should be sent through which path to maximize the gain (or minimize the distortion)

Q1. Allocation algorithm for LC

- Gain function
  - Gain = 0 if no layer or only enhancement layer is received
  - Gain = Δ if only base layer is received
  - Gain = 1 if both layers are received



- Want to maximize the expected gain

$$E(g) = \sum_{i=1}^{K} \left(\underbrace{\Delta\gamma_{L_i}(1-\gamma_{H_i})}_{\text{Gain obtained by receiving only base layer}} + \underbrace{\gamma_{L_i}\gamma_{H_i}}_{\text{Gain obtained by receiving both layers}} \right)$$

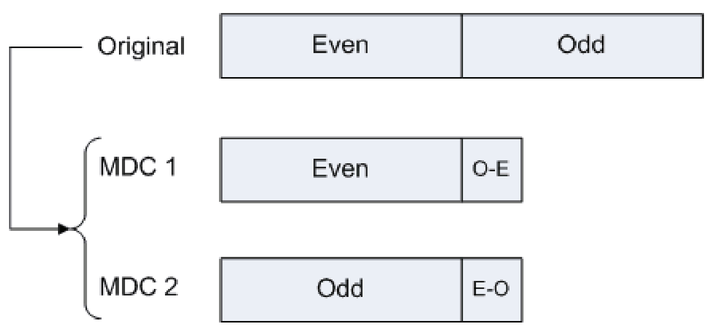
$$\gamma_u = 1 - p_1^{N(u,1)} p_2^{N(u,2)}$$

- Using dynamic programming to find optimal allocation

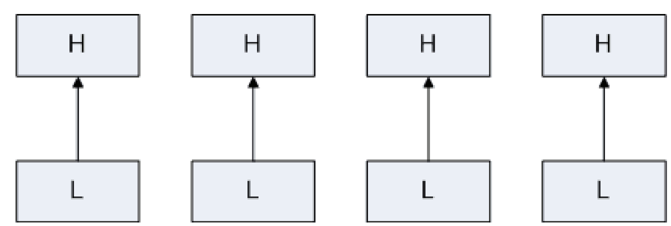
Q1. Experiment: Data

- Original data
  - PCM encoded audios, sampling rate 8kHz, 8 bits/sample
  - f116.wav: Female speech sentence, 21600 samples, 2.7s
  - CuckooWaltz.wav: birdsong in waltz tune, 32000 samples, 4.0s

- For every 800 original samples
  - MDC (Jiang's method, 2000)
    - MDC1 (inverse for MDC2)
      - Even: finer resolution (PCM, 8 bits/sample)
      - (Odd – Even): coarser (ADPCM, 2 bits/sample)
    - Packet size: 500 bytes (25% redundancy)

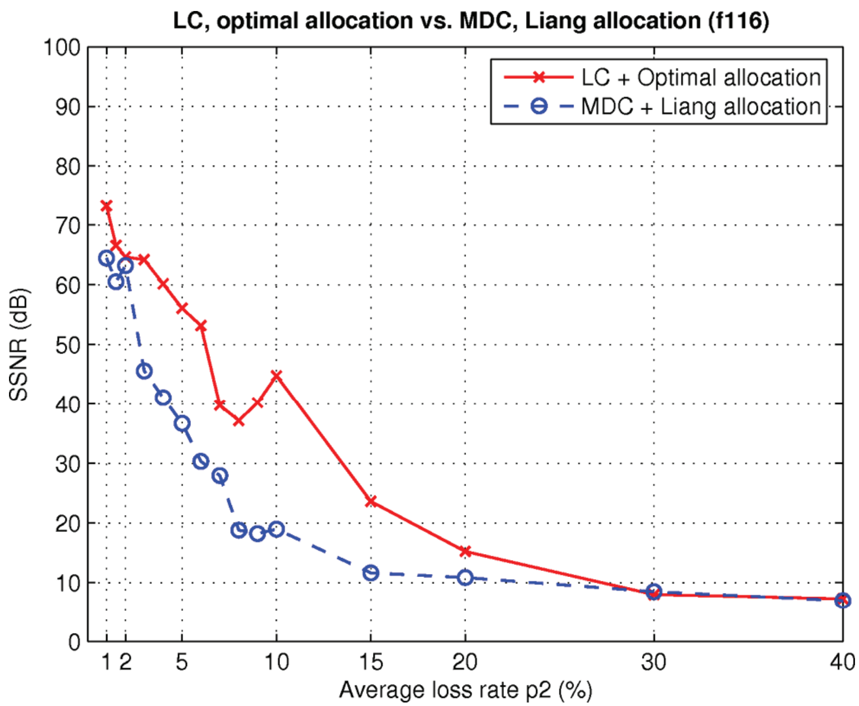


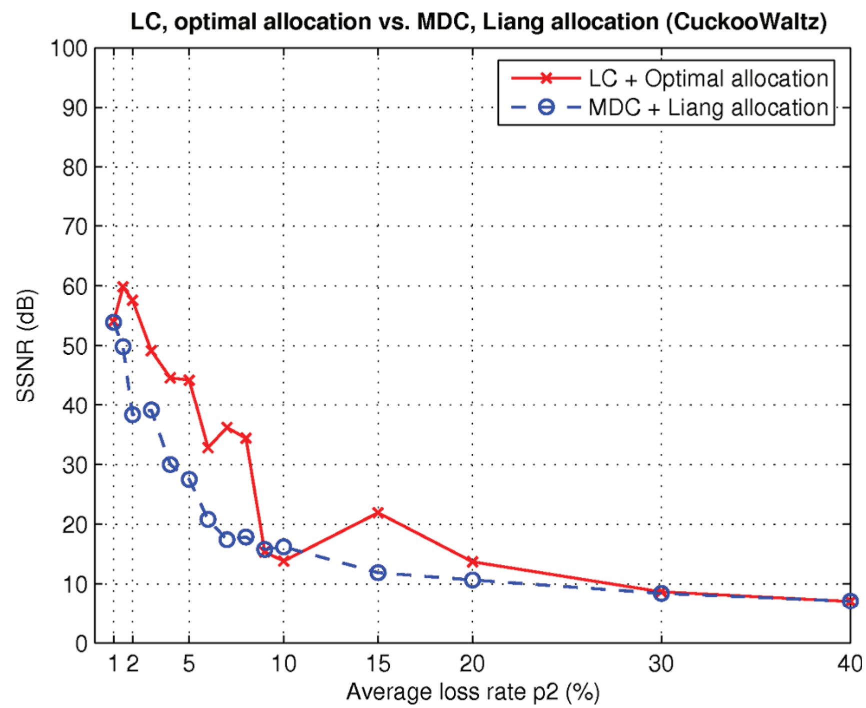
  - LC
    - Base-layer packet: 4 MSB
    - Enhancement-layer packet: 4 LSB
    - Packet size: 400 bytes
    - Compare SSNR of encoded packet vs. original
      ⇨ importance-level ratio between 2 packets in pair



Q1. Exp. result: $B_1 = B_2 = 5KB/s$

- MDC allocation
  - Liang algorithm: Each description over one path (Wang, 2002; Liang, 2001)



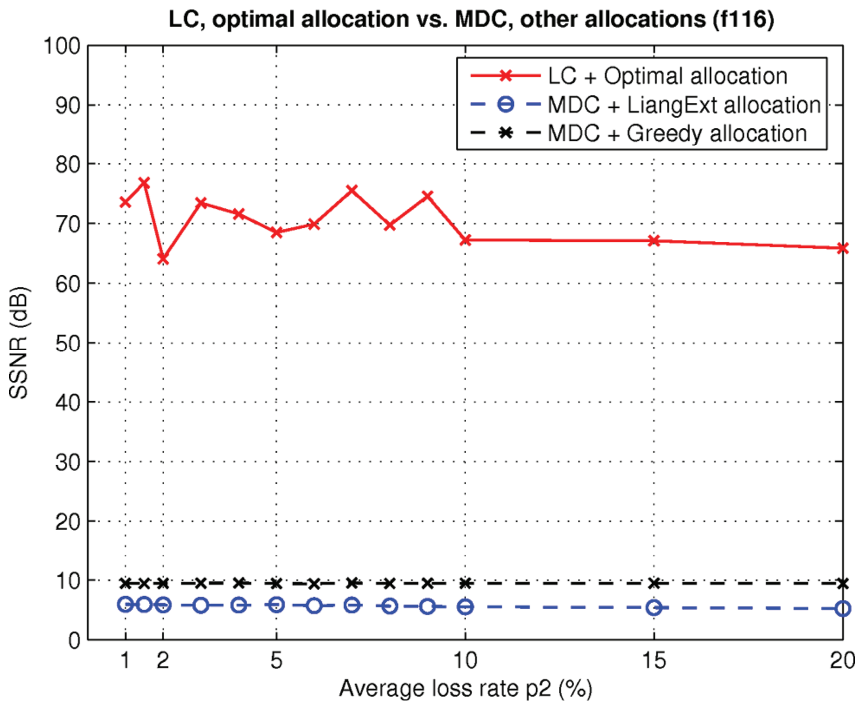


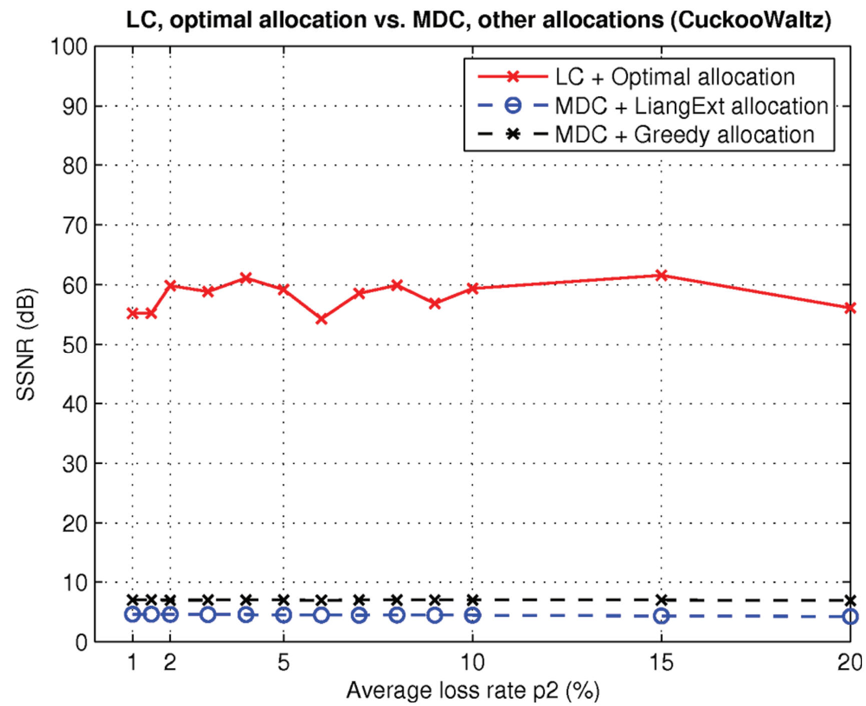
- Original: f116.wav
- At 5%:   MDC sample   **LC sample**
- At 10%: MDC sample   **LC sample**

- Original: CuckooWaltz
- At 5%:   MDC sample   **LC sample**
- At 10%: MDC sample   **LC sample**

Q1. Exp. result: B_1 = 8KB/s, B_2 = 2KB/s

- MDC allocation
  - LiangExt algorithm: Each description over one path until all bandwidth are used up
  - Greedy algorithm: From left to right, packets are sent over 2 paths until all bandwidth are used up



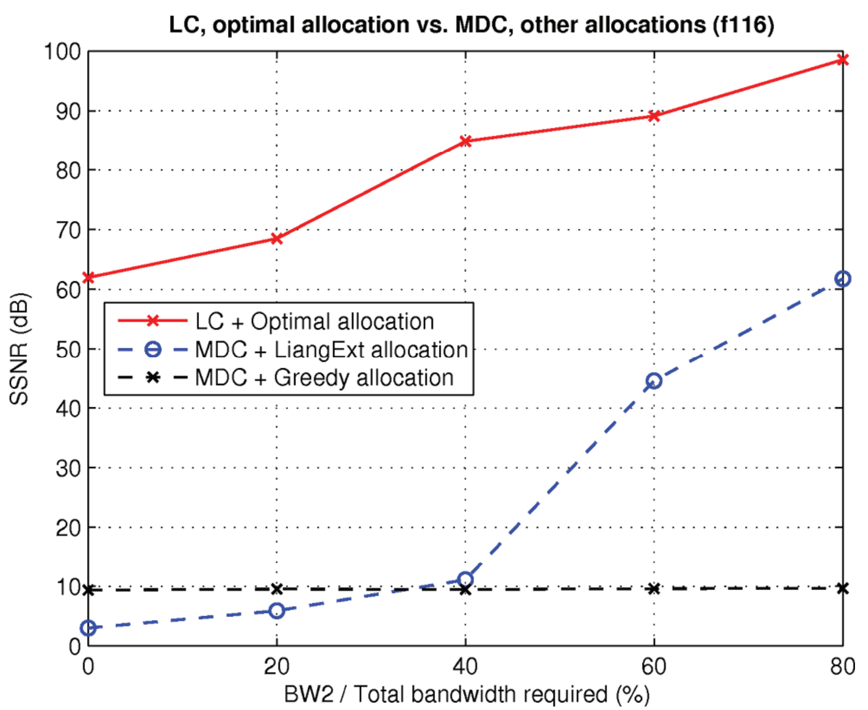


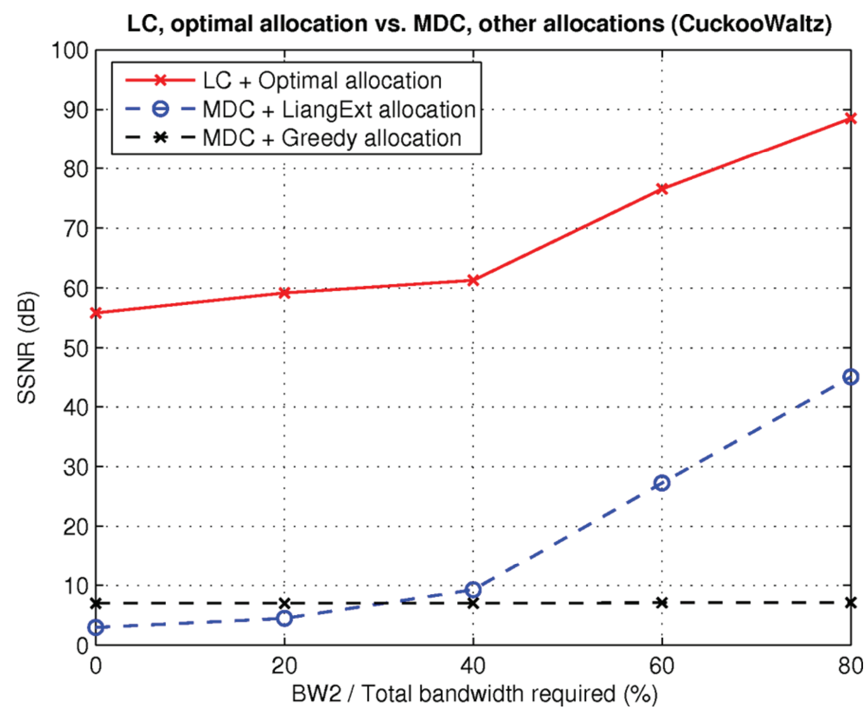
- **LC + Opt (at 5%): (55-64dB > MDC, Greedy)**
- MDC + LiangExt:
- MDC + Greedy: (3-5dB > MDC, LiangExt)

- **LC + Opt (at 5%): (48-55dB > MDC, Greedy)**
- MDC + LiangExt:
- MDC + Greedy: (2-3dB > MDC, LiangExt)

Q1. Exp. result: B_1 = 8KB/s, B_2 varies, p_2 = 5%

- MDC allocation
  - LiangExt algorithm: Each description over one path until all bandwidth are used up
  - Greedy algorithm: From left to right, packets are sent over 2 paths until all bandwidth are used up





- **LC + Opt.: 37-74dB > MDC, LiangExt**
- **LC + Opt.: 44-55dB > MDC, LiangExt**

Q1. Should we ***prioritize*** packets? – Summary

- If all packets are equal, we don't have much choices
  - Any MDC packet can be sent over any path
- If packets have different priorities, there is a room for improvement
  - We can decide which LC packet to send over which path
  - Optimal allocation of LC packets brings better quality

Roadmap

- Motivation
- The questions, our approaches & results
  - Should we ***prioritize*** packets?
  - ***What and how*** to prioritize packets?
  - How to ***schedule*** prioritized packets?
- Conclusion

Q2. ***What and how*** to prioritize packets?

- In many works, packets' priorities are
  - Based on their role in coding process
    - Syntax data: sequence header, slice header, frame type…
  - Equal for packets carrying same syntax data
  - Aimed to maximize objective quality: frame rate, PSNR…

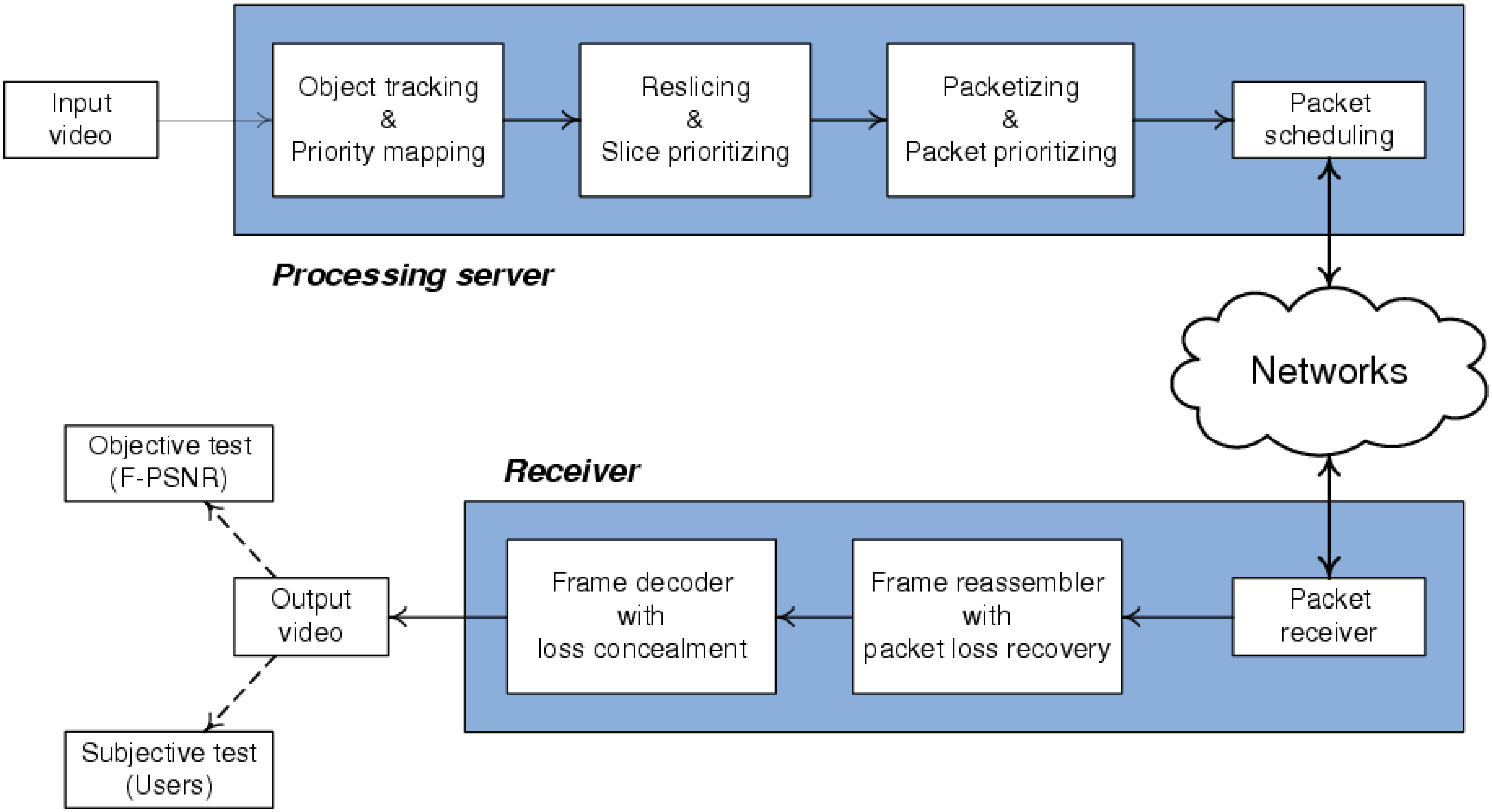
- However, users' interests are
  - Depending on applications, context, themselves
    - In news: audio >> video
  - Unequal for packets carrying same syntax data
    - In video surveillance: certain regions are more important
  - Aimed to maximize subjective quality: user experience, MOS

- Ultimately, users are the judges of any system

Q2. Prioritize based on ***semantic content***

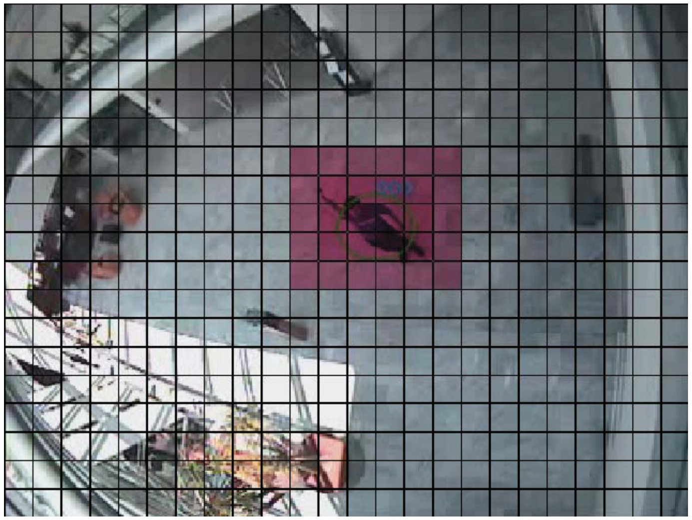
- Content-based encoding
  - MPEG-4 (object coding), JPEG 2000 (Region of Interest)
- Content-based bandwidth/bit allocation
  - Yang & Nahrstedt (2005): allocate more bandwidth to high-activity camera
- Different works, different “content”
  - Syntactic
    - slice headers, intra-coded MB (Pascal et al., 2001)
  - Semantic: scene, frame, region
    - Sport videos: event structure (Shih-Fu Chang et al., 2001)
    - Lecture videos: text regions (Liu et al., 2004)
    - Real-time videos: eye-tracking device (Komogortsev, 2004)

Q2. How to prioritize

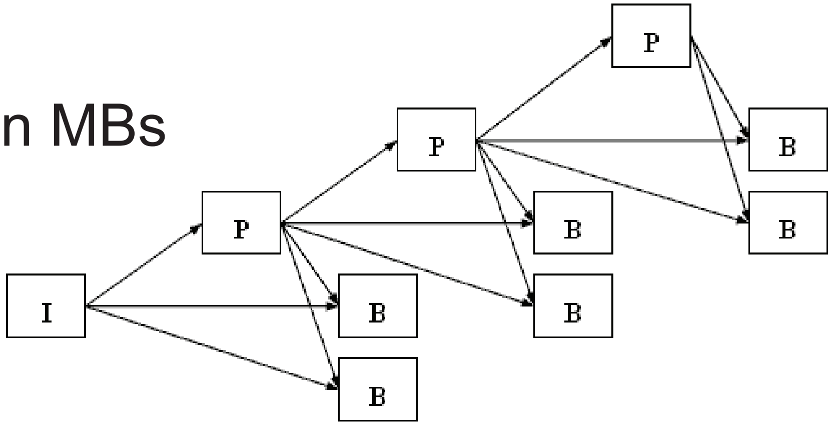


Q2. Priority mapping for macroblocks

- Effective priority of a MB (w_m)
  - Content priority (wc_m)
    - Blobs: Interest regions in frames
    - Protected regions = (1+k%) * blobs
      - Inside: $wc_m = 1$
      - Outside: $wc_m = w_{max} = 4$



  - Coding dependencies between MBs



  - $w_m = \max \{wc_{m'} \mid m' \text{ dependent on } m\}$

Q2. Reslicing, packetizing, prioritizing

- Slice
  - A group of consecutive macroblocks with same effective priority w_m
  - Size $\leq$ 1400 bytes
  - Effective priority of slice s: $w_s = w_m$

- Packet
  - Packetizing follows RFC2250 (RTP payload format for MPEG1/MPEG2)
    - SEQ, GOP, PIC header at beginning of payload
    - Size $\leq$ MTU
  - Effective priority = Content priority + Syntax priority
    - Content priority wc_u = average of its slices' priorities
    - Syntax priority ws_u
      - **Which syntax data?** SEQ, GOP, PIC, a combination of them
      - **Which should have higher priority?**

Q2. Experiment

- Content-based prioritization: [Blob + SEQ] scheme
  - Effective priority w_u = Content priority wc_u + Syntax priority ws_u
  - Packet with SEQ header: $ws_u = w_{max}$
- Frame-based prioritization: [PIC + SEQ] scheme
  - Effective priority w_u = Syntax priority ws_u
  - Packet of I, P, B frame: $ws_u = 2, 1, 0$, respectively
  - Packet with SEQ header: $w_u = ws_u + w_{max}$
- Want to compare [Blob + SEQ] and [PIC + SEQ]

Q2. Experiment: Implementation

- Object tracking
  - Modified *blobtrack* module in OpenCV 1.0 (ffmpeg + Dalí)
- Prioritizing (mnt + Dalí)
  - Process 1 GOP at a time
- Sending (stand-alone module for batch process)
  - Highest-priority packet are sent first
  - Reschedule a packet if it's lost (after an RTT)
  - Drop a packet if its playout deadline is over
- Receiving (mnt + Dalí)
  - Lost GOP, PIC header: Recover at reassembler
  - Lost MB or frame: Conceal at frame decoder
- Measuring
  - **Focused-PSNR:** PSNR of interest regions (mnt + ffmpeg)
  - **19 users:** MOS by ITU-R BT.500-11 (MSU Perceptual Video Quality tool)

Q2. Experiment: Parameters

- Two video surveillance videos from PETS benchmark datasets
  - pets2002-set1.mpg: 142 frames, 640x240, 25 fps (172KB/s)
  - Walk1-man.mpg: 200 frames, 384x288, 25 fps (124KB/s)

		PIC + SEQ	Blob + SEQ	Overhead
pets2002-set1	n_{RTP}	541	588	8.7%
	r	122,333	123,850	1.2%
Walk1-man	n_{RTP}	1058	1125	6.3%
	r	169,370	171,082	1.1%

- Network
  - Markov 2-state model: Ave. loss from 0 to 10%, excluding queue loss
  - RTT: 100, 200, 300ms
  - For each video, transmission rate = average data rate

Table 3.2: nRTP: # RTP packets, r: required sending rate (bytes/s, included headers)

Q2. Exp. result: pets2002-set1.mpg

- Content-based: [Blob + SEQ]

- Frame-based: [SEQ + PIC]

Play **Pause**

Sending rate = 172KB/s, Delay = 1s, Network loss rate = 5%, RTT = 100ms

Q2. Exp. result: Walk1-man.mpg

- Content-based scheme: [Blob + SEQ]

- Frame-based scheme: [SEQ + PIC]

Play **Pause**

Sending rate = 124KB/s, Delay = 1s, Network loss rate = 5%, RTT = 100ms

Q2. Exp. result: MOS & F-PSNR

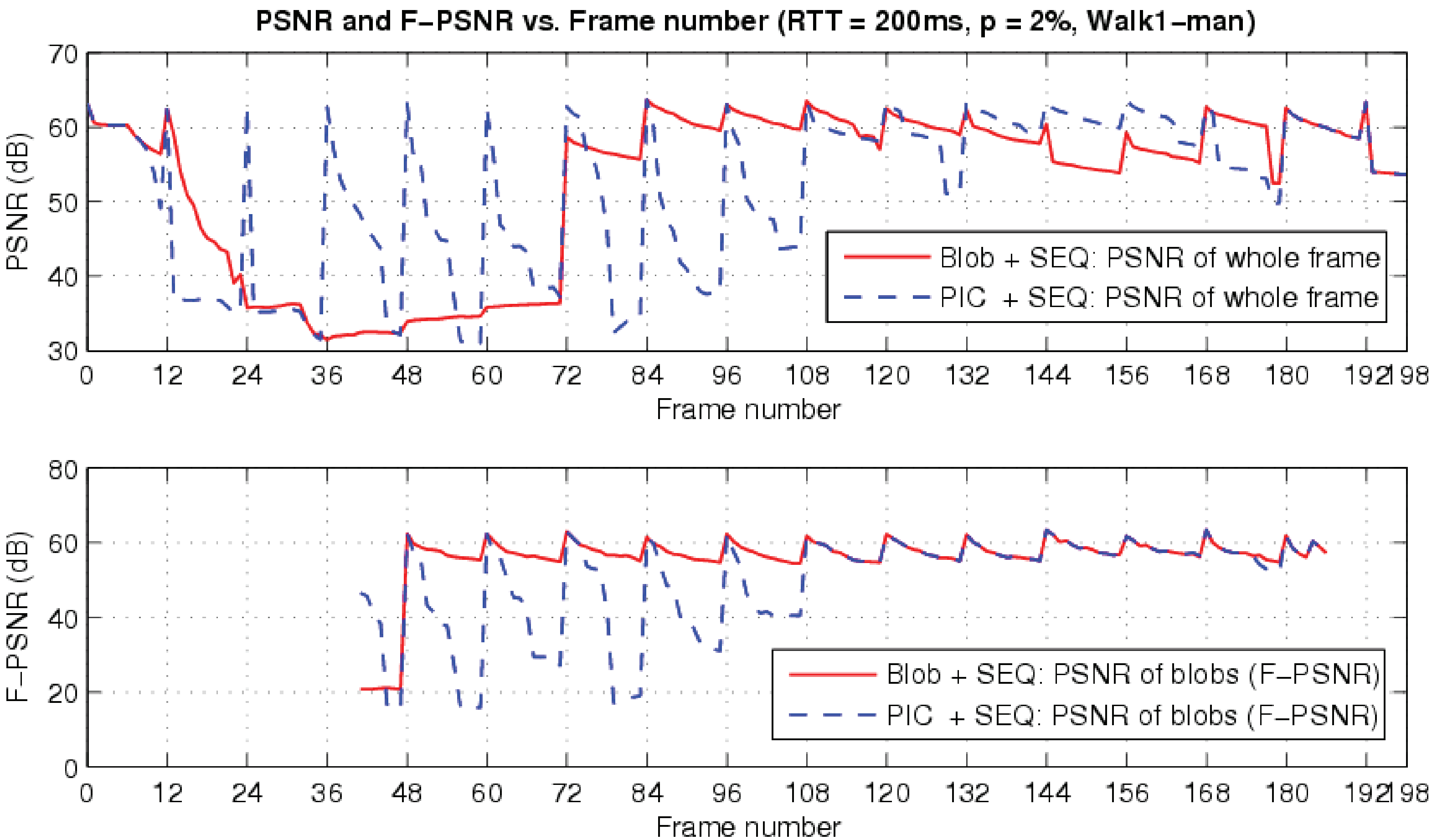
- Objective measurement

		Blob + SEQ		PIC + SEQ	
		MOS_a	σ	MOS_a	σ
pets2002-set1 (loss rate = 5%, RTT = 100ms)	Pair 1	**9.16**	1.20	**0.84**	1.20
	Pair 2	**9.05**	1.09	**0.95**	1.09
	Pair 3	**8.39**	1.53	**1.61**	1.53
Walk1-man (loss rate = 5%, RTT = 100ms)	Pair 1	**7.78**	1.88	**2.22**	1.88
	Pair 2	**9.13**	1.28	**0.87**	1.28
	Pair 3	**8.90**	1.81	**1.10**	1.81

- Subjective measurement

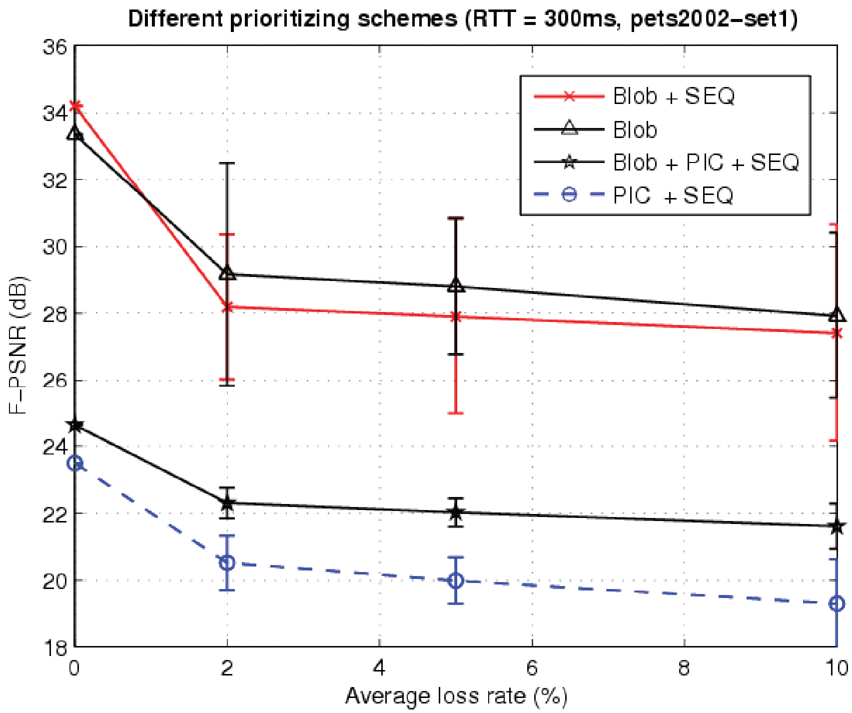
Video	Loss rate	F-PSNR improvement		
		100ms	200ms	300ms
pets2002-set1	0%	8.09	7.64	10.71
	2%	8.77	8.71	7.67
	5%	7.94	8.30	7.91
	10%	7.99	8.60	8.13
Walk1-man	0%	5.67	5.67	5.67
	2%	8.80	8.56	7.21
	5%	7.94	7.09	8.25
	10%	9.00	9.69	9.42

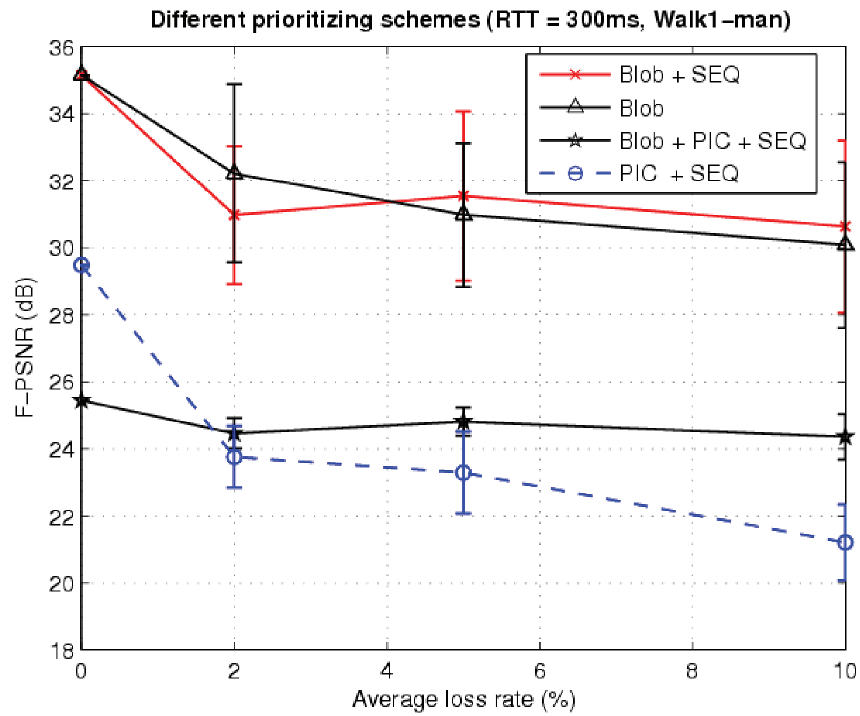
Q2. Exp. result: PSNR & F-PSNR



- Frame-based: Equal protection for all GOPs, with or without blobs
- Content-based: More protection for GOPs with blobs, blobs within frames

Q2. Exp. result: Simplicity is good





- [Blob] vs. [Blob + SEQ]: slightly higher
  - Insert SEQ/GOP header at every GOP + recovery if lost
  - [Blob + SEQ] protects packet with SEQ/GOP header of every GOP
- [Blob + SEQ] vs. [Blob + SEQ + PIC]: much higher
  - Adding frame-type priority creates confusion, e.g.,
    Priority of a High-content Low-syntax pkt = Priority of a Low-content High-syntax pkt

Q2. ***What and how*** to prioritize packets? – Summary

- Content-based prioritization is much better than Syntax-based prioritization
  - Objective: 6–11dB F-PSNR improvement
  - Subjective (MOS): 7.8–9.2 (content) vs. 0.9–2.2 (syntax)
- Using semantic content & syntax information
  - More priority to semantic content
  - Using lots of syntax information is not useful
- Focused-PSNR to measure regions' quality

Roadmap

- Motivation
- The questions, our approaches & results
  - Should we ***prioritize*** packets?
  - ***What and how*** to prioritize packets?
  - How to ***schedule*** prioritized packets?
- Conclusion

Q3. How to ***schedule*** prioritized packets?

- One feature
  - Extensively studied in operating system, buffer scheduling…
  - Highest-priority first: **FirstFit** (Chang et al., 2001)
  - Earliest-deadline first: EDF, **Urgent**
  - Not enough
- Many features
  - Chakareski et al. (2003): priority (dependency, role in concealment), deadline + RTT, loss rate, bandwidth
  - Difficult to pass across layers/networks, to process
- Our proposal: Using some common & simple features
  - Priority & deadline: **GenFlag2** (modified, Chrobak et al., 2004)
  - Priority, deadline + RTT: **GenFlagNet**, **EoH**

Q3. GenFlag2

- Alternatively sends e'_t and h_t
- Favors e'_t
  - If $e'_t = h_t$: send e'_t next time
  - If this is the h_t chance, but e'_t has certain priority & urgent deadline, then send e'_t
- α = 7/11, β = 8/11

Set $eFlag = false$.

At every sending opportunity t, do the following.

1. Update the set of pending packets Q_t.
2. Find the highest-priority packet h_t (use tie-breaking rule in Section 4.3.2 if necessary).
3. Find the earliest-deadline packet $e_t^{'}$ among the packets j whose $\boxed{v(j) \geq \alpha v(h_t)}$ (use tie-breaking rule in Section 4.3.3 if necessary).
4. If $eFlag = false$

   then schedule $e_t^{'}$

   $\boxed{\text{If } e_t^{'} \neq h_t \text{ then set } eFlag = true}$

   Else

   Set $eFlag = false$

   $\boxed{\begin{array}{l}\text{If } [t + sz(e_t^{'})/R \leq dl(e_t^{'}) < t + 2sz(e_t^{'})/R] \text{ and } [v(e_t^{'}) \geq \beta v(h_t)] \\ \quad \text{then schedule } e_t^{'}\end{array}}$

   Else schedule h_t

Q3. EoH (Earliest or Highest)

- Send e_t if h_t have K chances to be sent
- $1 - p^K \geq p_o$
  - probability of successful receiving: p_o
  - packet loss probability: p

Set K the number of times that the scheduler wishes to send the highest-priority packet h_t.

At every sending opportunity, say at the time t, do the following.

1. Update the set of pending packets Q_t (remove packets with deadline later than t and add new coming packets).
2. Find the highest-priority packet h_t (use tie-breaking rule in Section 4.3.2 if necessary).
3. Find the earliest-deadline packet e_t. (use tie-breaking rule in Section 4.3.3 if necessary).
4. If $\left(t + \frac{sz(e_t)}{R}\right) < dl(h_t) - (K-1)\left[W + \frac{sz(h_t)}{R}\right]$

   then schedule e_t

   Else

   schedule h_t

Q3. GenFlagNet

- GenFlag2 vs. EoH
  - GenFlag2: h_t and e'_t
  - EoH: h_t and e_t with RTT consideration

- GenFlagNet
  - h_t and e'_t with RTT consideration

Q3. Experiment

- Highest-priority first > Earliest-deadline first?
  - FirstFit vs. Urgent
- Priority + deadline > Priority or deadline?
  - GenFlag2 vs. FirstFit and Urgent
- Priority + deadline + RTT > Priority + deadline?
  - GenFlag2 vs. GenFlagNet: RTT effects
  - GenFlagNet vs. EoH: packet-selection effects

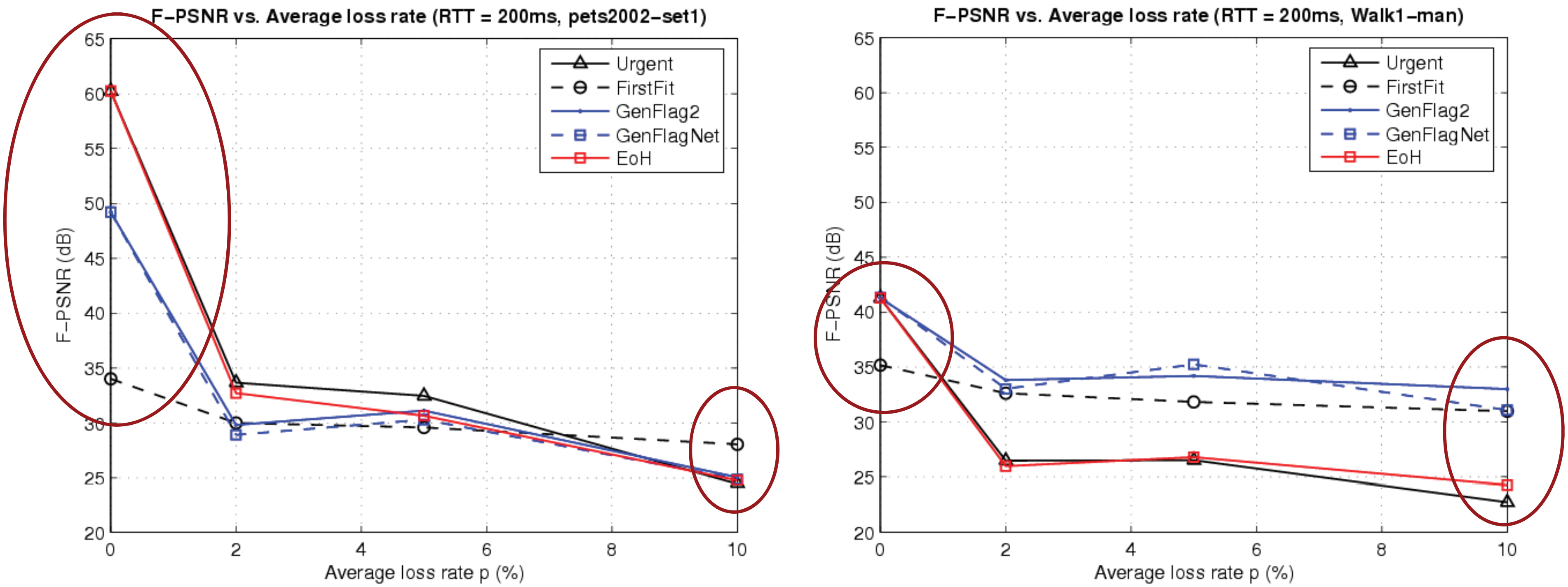
Q3. Experiment: Parameters

- Two video surveillance videos from PETS benchmark datasets
  - pets2002-set1.mpg: 142 frames, 640x240, 25 fps (172KB/s)
  - Walk1-man.mpg: 200 frames, 384x288, 25 fps (124KB/s)
  - Content-based prioritized by [Blob + SEQ] scheme

- Network
  - Markov 2-state model: Ave. loss from 0 to 10%, excluding queue loss
  - RTT: 100, 200ms
  - Transmission rate: 80–140% average data rate

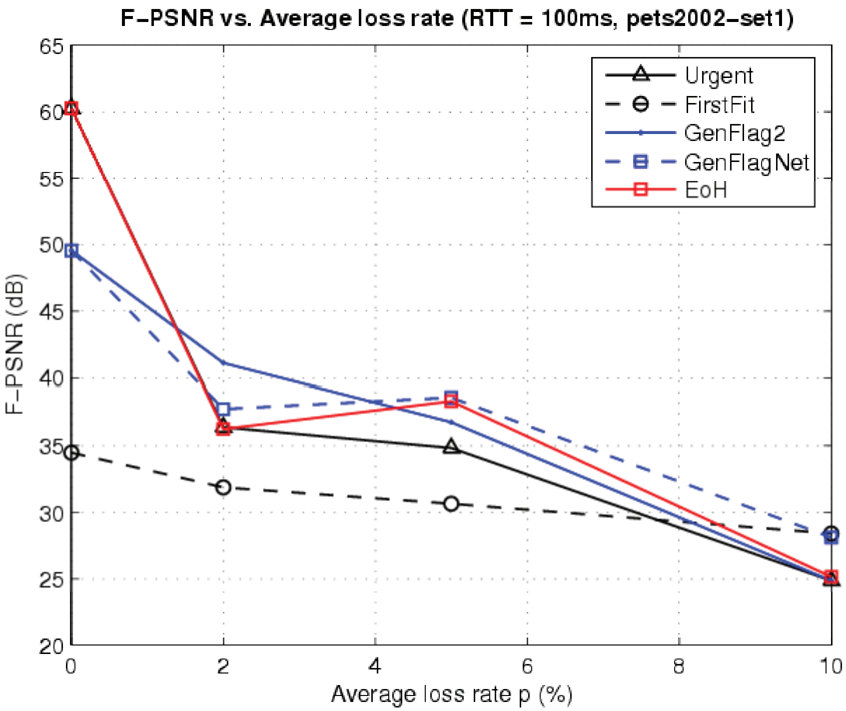
- Measurements:
  - Average PSNR, F-PSNR of 15 runs for each network configuration

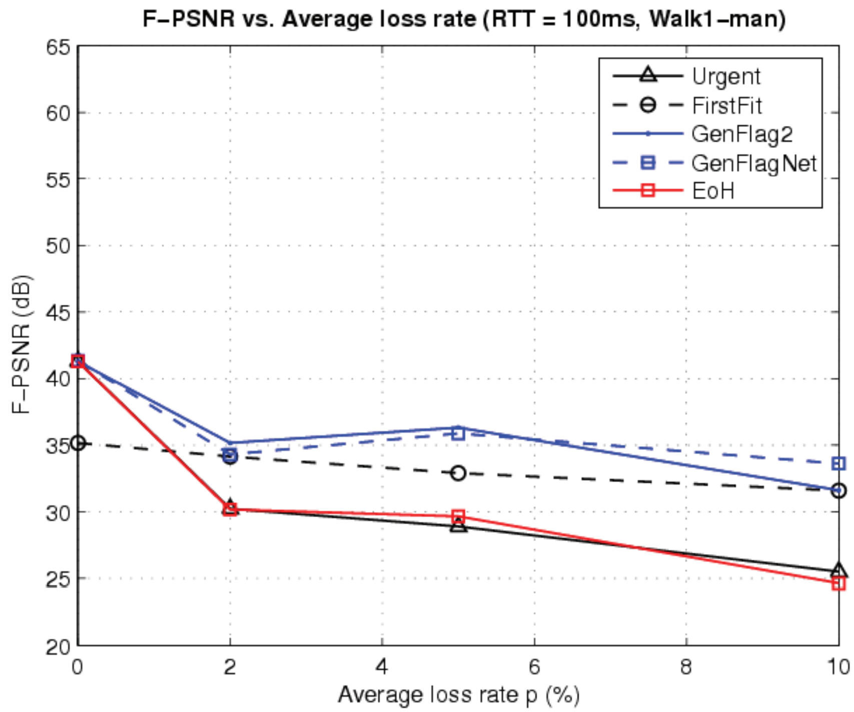
Q3. Exp. result: FirstFit vs. Urgent



- Neither is good for all situations
  - 0% network loss: Urgent always better
    - *Pets2002-set1*: 26dB in F-PSNR, 19dB in PSNR
    - *Walk1-man*: 6dB in F-PSNR, 5dB in PSNR
  - 10% network loss: FirstFit always better
    - *Pets2002-set1*: 4dB in F-PSNR, 2-4dB in PSNR
    - *Walk1-man:* 6-8dB in F-PSNR, < 1dB in PSNR

Q3. Exp. result: GenFlag2 vs. FirstFit vs. Urgent





- 0% network loss: Urgent ≥ GenFlag2 >> FirstFit
- Other loss ratios
  - GenFlag2 = max{Urgent, FirstFit} + (2–4 dB in F-PSNR)

- GenFlag2: not the best in all scenarios, but reasonably good
  - GenFlag2 ≈ max{Urgent, FirstFit}

Q3. Exp. result: GenFlag2 vs. GenFlagNet vs. EoH

- Effect of RTT: GenFlag2 vs. GenFlagNet
  - Very similar performance (~1dB difference)
  - Considering RTT doesn't help much
- Effect of packet selection: GenFlagNet vs. EoH
  - 0% network loss
    - EoH always better (41–60dB, GenFlagNet ~ 41–49dB)
  - Other loss rates
    - GenFlagNet better in F-PSNR, EoH is better in PSNR
- Experiments with different transmission rates give similar results

Q3. How to ***schedule*** prioritized packets? – Summary

- No single best scheduler
  - Bad conditions (high loss, low bandwidth)
    - Highest-priority first (FirstFit) is the best
  - Normal conditions (loss < 10%, average bandwidth)
    - Priority & deadline (GenFlag2) is the best
    - Adding RTT doesn’t help much (GenFlagNet)
  - Good condition (zero loss or high bandwidth)
    - EoH is the best, but GenFlag2 is quite close

- FirstFit is rather stable w.r.t of RTT & loss rate

- GenFlag2 is good enough for most situations

Roadmap

- Motivation
- The questions, our approaches & results
  - Should we ***prioritize*** packets?
  - ***What and how*** to prioritize packets?
  - How to ***schedule*** prioritized packets?
- Conclusion

Conclusion

- Should we ***prioritize*** packets?
  - Packets often have different priorities, from coding & user perspectives
  - With prioritization, we can optimize delivery to achieve better quality

- ***What and how*** to prioritize packets?
  - Users know what semantic content is important
  - Prioritization should reflect their needs
  - Content-based prioritization >> syntax-based prioritization

- How to ***schedule*** prioritized packets?
  - Delivering prioritized packets is not trivial
  - Should consider both priority and deadline
  - Considering RTT doesn't help much

Conclusion

- In streaming multimedia packets, better results could be achieved if we prioritize them (especially based on users' interests), and send them based on their priority and deadline.

- Future works
  - How systems can monitor and adapt to users' needs?
    - In the same context, different user wants different things
  - For multiple streams, how to measure perceived quality?
    - Average quality among streams? Sum? Weighted sum?
    - Overall experience?

Thank you…

Thank you very much and indeed!