RoSE Technical Report - II

—

Performance Evaluation of MDC

Phase II

I. Introduction

This report presents the second phase of performance evaluation of MDC schemes in the Robust Streaming Environment (RoSE) project [1]. The spatial 2MD scheme is studied. First, the performance is evaluated using PSNR metric under perfect network conditions. This baseline scenario provides the fundamental overhead of integrating MDC into a multimedia streaming system. Then, in a more realistic setting, transmission impairments are considered on a network testbed. Other issues such as increased complexity are also crucial but not considered in this phase.

II. Spatial 2MD Scheme

A typical and low-cost way to produce multiple descriptions is to partition the source data into several sets and then compress independently to produce descriptions. The separation can be into odd- and even-numbered samples [2]. In the spatial dimension, this corresponds to spatial sampling of frames into N subsets. For $N = 2$, two balanced descriptions can be generated by separating odd/even lines [3]. This technique is denoted as *spatial 2MD* and illustrated in Figure 1 [4].

In spatial 2MD, each frame in the input video is separated into odd and even subframes at the Remux module of RoSE. The odd and the even subframes contain the odd and even lines, respectively. Therefore, the height of the frames are halved but the width does not change as shown in Figure 1. These two descriptions are encoded with half the bitrate of the original stream to keep the total bitrate constant. Then these descriptions are offline-processed/streamed and merged at the Postmux to reconstruct the received video. All these MDC-related operations are performed in the data plane. This property achieves compatibility with any incumbent codecs.

III. Performance Evaluation and Methodology

The MDC-enabled system is evaluated using experiments with various test videos and encoding bitrates. The relevant system parameters are listed in Table I. In the experiments, *Akiyo*, *Foreman*, *Hall*, *Mobile*, and *News* test sequences are used as the input video. They have 300 frames and CIF size. The original raw video is encoded into MPEG4 with varying bitrates between 48 kbps and 8192 kbps. The frame type structure is IPPPPPPPPPPP (no

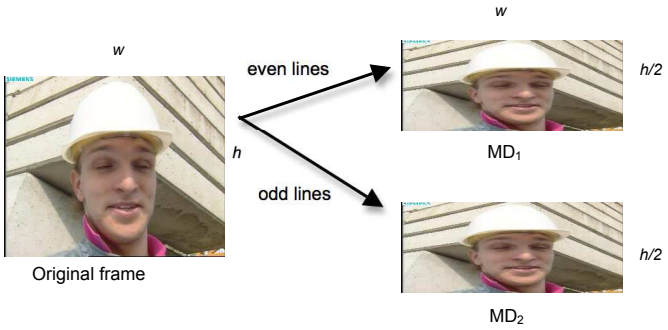


Fig. 1. Spatial 2MD coding scheme.

B frames). Then the raw video (scenario 1) or this encoded sequence (scenario 2) is fed into the Remux module and processed to generate descriptions in MPEG4 format. MPEG4 is preferred since it is one of the state-of-the-art coding formats. However, any codec can be employed as long as both Remux and Postmux modules support it. This multiple description coded video is encapsulated in a *.nut* file to be streamed. Subsequently, Postmux processes these descriptions to reconstruct the original video at the receiver side. This experimental setup is depicted in Figure 2.

Scenario 1 (the raw video input case) corresponds to streaming or content delivery systems where the multimedia can be processed beforehand in an offline fashion to take care of multiple description transmission. Moreover, it constitutes the baseline scenario for MDC based streaming. In scenario 2, the Remux module acts as a multiple description transcoding engine where the input is an already coded and compressed video. This may occur when MD is integrated into a network as an plug-in edge device for interfacing heterogeneous systems such as wireless access networks. Obviously, this setup also entails some drawbacks, the most notable being that the MDC performance will be “constrained” by the quality degradation caused by the prior encoding. Additionally, real-time transcoding comes with processing and delay overhead. However, cache-based delivery systems such as personalized TV or radio can benefit from this configuration since the delay burden is not a major issue. The overhead is also acceptable when various quality levels (distortions) are acceptable and distinguishable and the reconstructions produced at decoders (receivers) should be more valuable than nothing [2].

The Peak-Signal-to-Noise-Ratio (PSNR) metric is used to assess the penalty of the method and the quality of the MDC video. PSNR is one of the simplest and most widely used quality metrics. PSNR is calculated with the mean squared error (MSE), computed by averaging the squared intensity differences of distorted and reference frame pixels, along with the related quantity of PSNR. These are appealing because they are simple to calculate, have clear physical meanings, and are mathematically convenient in the context of optimization. The simplest implementation of this concept is the MSE, which objectively quantifies the strength of the error signal. But two distorted images with the same MSE may have very different types of errors, some of which are much more visible than others.

The frame (image) quality is measured as the MSE value which is defined as

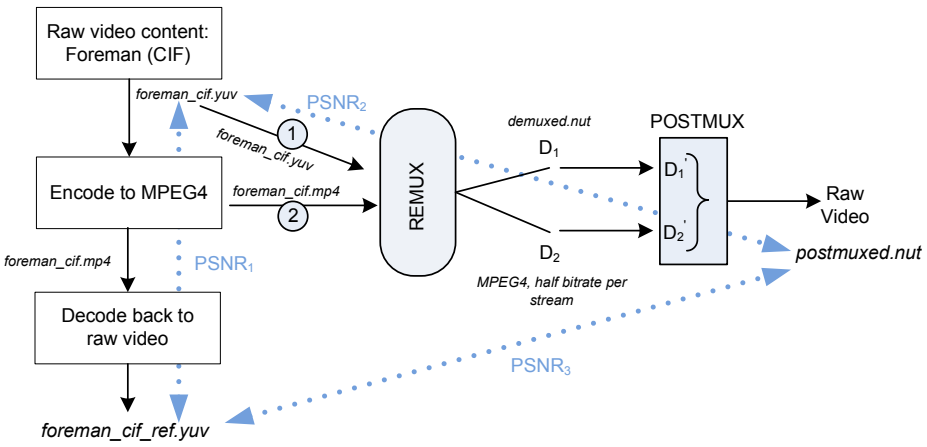


Fig. 2. Experimental setup and performance evaluation.

TABLE I

SYSTEM AND SIMULATION PARAMETERS

Parameter	Value
Channel Model	1: No impairments, 2: network emulation
Input raw videos	*Akiyo*, *Foreman*, *Hall*, *Mobile*, *News*
Input properties	YUV 420
Input codec	1: RAW, 2:MPEG4
Bitrate (kbps)	48, 96, 192, 384, 768, 1536, 2048, 4096, 8192
No. frames	300
Frame size	CIF
MPEG4 frame order	IPPPPPPPPPPP
MDC codec	MPEG4 Simple
MDC scheme	Spatial 2MD

$$MSE = \frac{1}{N^2}\sum_{i}^{N}\sum_{i}^{N}\left[X(i,j) - \widehat{X}(i,j)\right] \tag{1}$$

where, N^2 is number of pixels in image, $X(i,j)$ and $\widehat{X}(i,j)$ are the pixel values of the reference frame and of the final frame reconstructed from the received multiple descriptions, respectively. We use the following PSNR metric which is

$$PSNR(dB) = 20\log_{10}\frac{2^n - 1}{RMSE} \tag{2}$$

where p is the largest possible value of the signal (n = 8, i.e. $2^n - 1 = 255$ for grayscale images), and RMSE is the Root Mean Square Error between the two images given above, respectively.

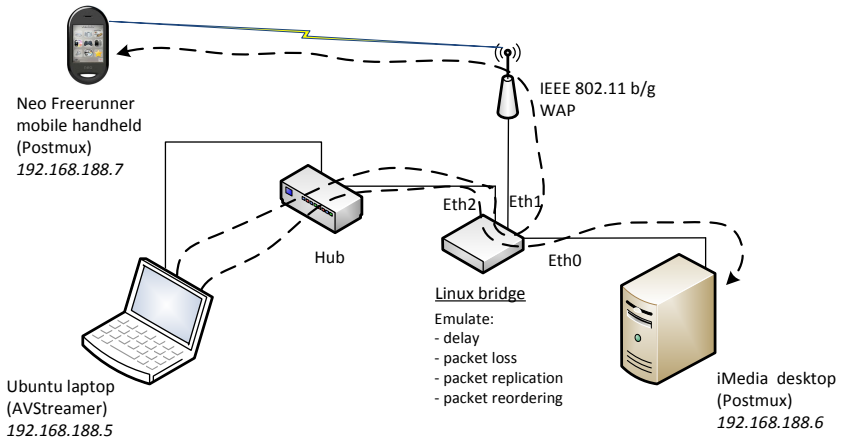


Fig. 3. RoSE testbed implementation.

A. RoSE testbed

For more realistic evaluation, the system is also tested over a testbed using network emulation for packet loss in addition to offline-processing. The testbed implementation is shown in Figure 3. It consists of linux hosts acting as server, client and network bridge. Additionally, wireless transmission is implemented using a IEEE 802.11b/g network with a linux-based handheld device as the end-point.

For network emulation, a linux bridge is used with `brtcl` and `net:netem` tools installed. This box has the capability to emulate delay, packet loss, replication, and reordering. The packet loss feature can be configured to emulate random packet loss following a statistical distribution or burst losses typical in wireless channels.

For combating packet loss, Postmux applies error concealment for the missing data such as motion vectors. In case where the entire frame is lost, repeating of previous frame is used.

TABLE II
ROSE TESTBED HOSTS

Host	iMedia (Postmux: Client)	Ubuntu (AVStreamer: Server)
OS	Fedora Release 10	Ubuntu 9.04
Kernel	2.6.27	2.6.28
Memory	501.1 MiB	497.2 MiB
Processor	Intel Pentium 4 2.40 GHz	Intel Pentium M 1.40 GHz

IV. EXPERIMENTAL RESULTS

In this section, we first present the results for the baseline scenario. The rate-distortion curves for Foreman sequence are plotted in Figure 2. Three cases are considered for PSNR evaluation. These measurements are marked as PSNR_i on Figure 2. For the time being, scenario 2 is not considered. In the first metric, PSNR_1, the distortion

due to encoding in MPEG4 is measured. This case is also equivalent to a single sender and receiver transmission without MDC by sending all the video packets using a single route under perfect channel conditions. This is plotted as the dashed line in Figure 4. In the second case, we measure the distortion cumulatively due to both MPEG4 and the subsequent MDC stage (MPEG4 decoding/encoding and MDC operations). This corresponds to PSNR_2 and shown as the solid line in Figure 4. And in the final case, PSNR_3 measures the distortion due to MDC. This is the most significant metric for focusing solely on the effect of MDC.

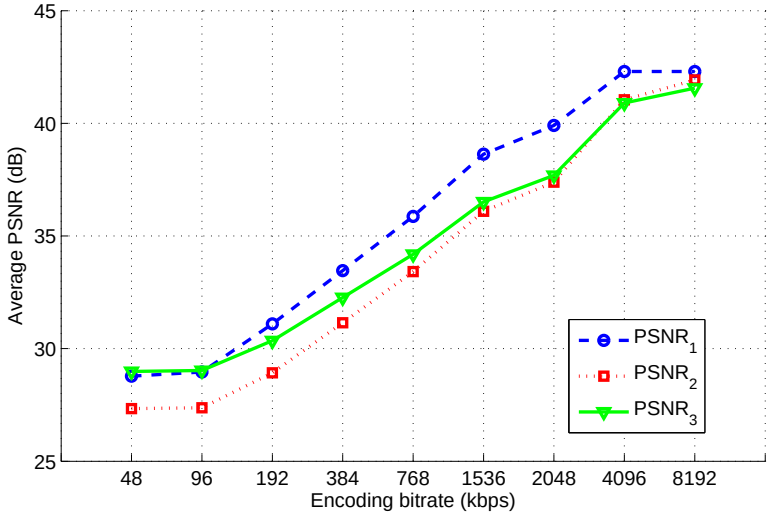
The average PSNR values for varying bitrates using raw video inputs and MPEG4-coded output at the Remux module are listed in Table III. Specifically, the average PSNR values for *Foreman* sequence from this table are plotted in Figure 4(a). For this sequence, the PSNR_1 value is about 28.96 dBs for 96 kbps compared to 42.30 dBs for 8192 kbps. The PSNR_2 has smaller values but follows a similar trend: about 27.37 dBs for 96 kbps compared to 41.93 dBs for 8192 kbps. The PSNR_3 value increases more sluggishly and is about 29.03 dBs for 96 kbps compared to 41.56 dBs for 8192 kbps. All PSNRs monotonically increase for increasing bitrates. The PSNR degrades substantially for low bitrates in all cases. As the bitrate increases, the difference between encoded-decoded MPEG4 output and original video fades since, for the high bitrates, single compression-decompression affects the quality of the video marginally. Therefore the gap between PSNR_3 and PSNR_2 closes and PSNR_3 converges to PSNR_2 (i.e., the MDC-free encoded-decoded sequence is almost identical to the original raw sequence in that case). These objective metrics match the actual quality of the videos because an obvious difference between qualities can be detected when the sequences are visually evaluated: the perceptual quality improves for increasing bitrates as seen in Figure 7 and 8. These trends are valid for all of the input sequences.

The average PSNR values for *Mobile* sequence is also shown in Figure 4(b),. For the other four sequences, we observe the *S-curve* showing an initiation phase, then fast growth and a final steady-state behavior in PSNR values as listed in III. However, this curve has not reached the steady-state region and ends up with a lower PSNR value than the other sequences. The reason is that this sequence has very rich texture with camera movement. If the encoding bitrate is increased beyond 8192k, the average PSNR will increase and then the saturation will be observed around the same PSNR values as the other sequences.

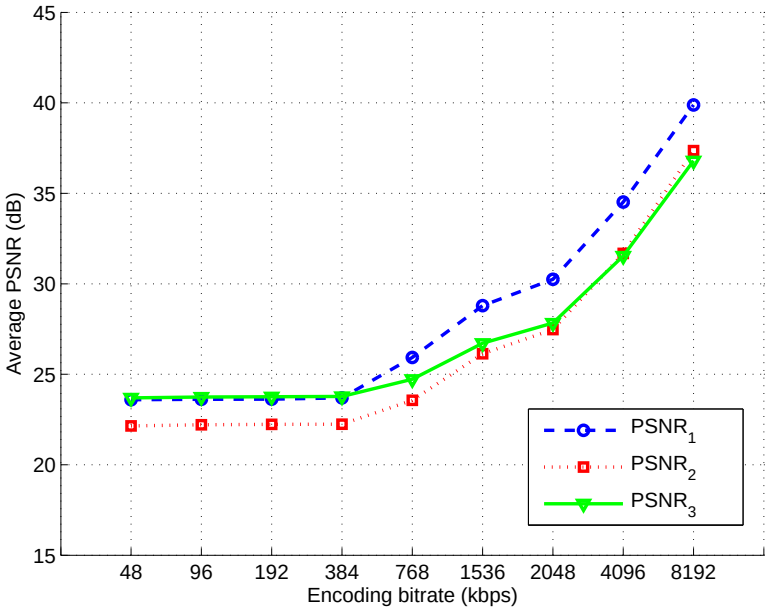
We also investigate the difference between PSNR_1 and PSNR_2, denoted as δ and defined as

$$\delta = PSNR_1 - PNSR_2 \tag{3}$$

It represents the pure PSNR overhead or penalty for integrating MDC in the transmission chain. Because the experienced PSNR would be PSNR_1 without MDC whereas it is degraded to PSNR_2 with MDC. This metric is dependent on the the characteristics of the input sequence and thus shows diverse behavior. The experimental results are shown in Figure 6. For Foreman sequence, δ values start from 1.59 for 96 kbps and increases up to 2.53 at it peak for 1536kbps. Then, it drops down to 0.37 for the last bitrate. This is expected since both metrics suffer from extreme degradation for very low bitrate and almost lossless compression for very high bitrate. On the contrary, However, all the sequences except *Mobile* shows one common trait: the final δ is always smaller than the first



(a) *Foreman* sequence (300 frames and CIF size.)



(b) *Mobile* sequence (300 frames and CIF size.)

Fig. 4. Average PSNR values for varying bitrates using raw video input and MPEG4-coded output at the Remux module (scenario 1).

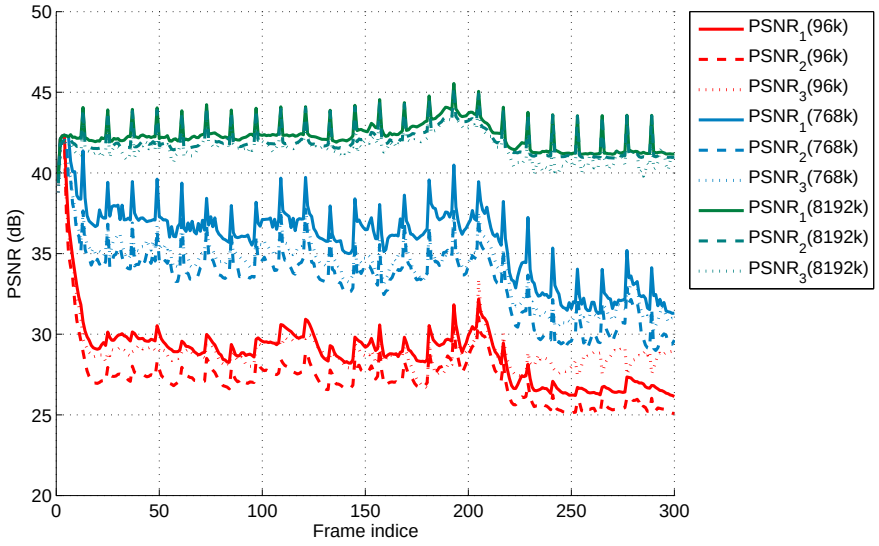


Fig. 5. PSNR performance for the entire sequence with encoding bitrates 96 kbps (red), 768 kbps (blue), and 8192 kbps (green). Solid lines are for $PSNR_1$, whereas dashed and dotted lines represent $PSNR_2$ and $PSNR_3$, respectively.The raw video is *Foreman* sequence with 300 frames and CIF size.

δ. Again, *Mobile* sequence needs to reach higher bitrates to complete its evolution to a stable point. It will also decrease to a lower value for bitrates larger than 8192k. The average value (μ_δ) over test sequences ranges between 1.06 (for 8192kbps) and 2.09. It peaks in the midrange of the bitrates. Thus, it shows a consistent behavior with a parabolic typology. The standard deviation (σ_δ) ranges between 0.56 and 0.93. These metrics are also plotted in Figure 6 with dashed and dotted lines, respectively.

The same trend is also apparent in Figure 5. In this figure, PSNR performance for the entire sequence with encoding bitrates 96 kbps (blue), 384 kbps (red), and 8192 kbps (green) are shown for representing the general behaviour. Other bitrates are omitted for the sake of brevity. Solid lines are for $PSNR_1$, whereas dashed and dotted lines represent $PSNR_2$ and $PSNR_3$, respectively. All PSNR values behave as expected and degrade for decreasing encoding bitrates. We also observe the diminishing returns for increasing the bitrate to very high bitrates. The PSNR gain becomes much less sensitive to increasing bitrate for high bitrates (the gap closes between consecutive bitrates.) As the limit case, when the compression is omitted (the bitrate passes the raw video bitrate), the gain from increasing the bitrate will be zero. Also, a periodicity is observed due to independent-coded I-frames in the sequence.

It is an interesting fact that for low bitrates such as 96k, when the compression is repeated, the psnr values slightly change (very minor variations in less than 10 percent of results). However, for large bitrates such as 8192k, the results are exactly reproducible.

For visual evaluation, we provide sample frame captures from original, remuxed and postmuxed video sequences

TABLE III
AVERAGE PSNR VALUES FOR TESTED SEQUENCES

Sequence	PSNR type	48k	96k	192k	384k	768k	1536k	2048k	4096k	8192k
Akiyo	1	35.96	38.68	41.03	43.46	45.07	45.07	45.07	45.07	45.07
	2	33.21	35.82	38.26	40.96	43.57	44.19	44.19	44.19	44.19
	3	34.05	36.47	38.85	41.54	44.24	44.76	44.76	44.76	44.76
Foreman	1	28.77	28.96	31.10	33.46	35.87	38.63	39.90	42.30	42.30
	2	27.34	27.37	28.93	31.15	33.42	36.10	37.39	41.05	41.93
	3	28.99	29.03	30.35	32.27	34.19	36.51	37.69	40.91	41.56
Hall	1	30.64	33.41	35.63	37.32	39.03	40.83	41.73	42.80	42.80
	2	29.05	31.77	34.20	36.26	38.05	39.70	40.56	42.12	42.12
	3	29.97	32.47	34.95	37.08	38.90	40.28	41.20	42.67	42.67
Mobile	1	23.59	23.62	23.63	23.70	25.93	28.79	30.25	34.52	39.88
	2	22.15	22.21	22.23	22.24	23.56	26.13	27.47	31.68	37.36
	3	23.69	23.74	23.76	23.78	24.73	26.71	27.83	31.54	36.80
News	1	30.31	32.96	35.76	38.56	41.70	44.12	44.12	44.12	44.12
	2	28.27	30.36	33.09	35.93	39.12	42.53	43.28	43.28	43.28
	3	29.06	30.95	33.45	36.23	39.46	42.90	43.50	43.50	43.50
Average	$\overline{1}$	29.86	31.53	33.43	35.30	37.52	39.49	40.21	41.76	42.83
	$\overline{2}$	28.00	29.51	31.34	33.31	35.55	37.73	38.58	40.46	41.77
	$\overline{3}$	29.15	30.53	32.27	34.18	36.30	38.23	39.00	40.68	41.86

in Figure 7 and 8. These captures exhibit the positive correlation between the quality and encoding bitrate in the system. The performance loss due to MDC stage (remux+postmux) is hard to perceive with visual inspection. The significant quality degradation in all of the different cases for low bitrate encoding is to be noted.

A. Case 2: Transmission with Network Impairments

In the previous part, the baseline results for the were presented. As the second case, we provide the experimental results under emulated network impairments on RoSE testbed.

1) Single Packet Drop: In this experiment, a minimal distortion on video stream is realized using a single RTP packet drop. The Remux output is encoded at 1536 kbps and streamed to Postmux with a modified network API call implementing packet loss (packet-hijacking). The 73th packet during the transmission is dropped and the results are examined. All the streamed frames could be reconstructed at the Postmux, i.e. number of reconstructed frames is 300. The PSNR performance is given in Figure 9. In summary, the effect of impairment can be distinguished in the graph and also in the overall average PSNRs. There is a quality loss of 0.43 dB in terms of average PSNRs.

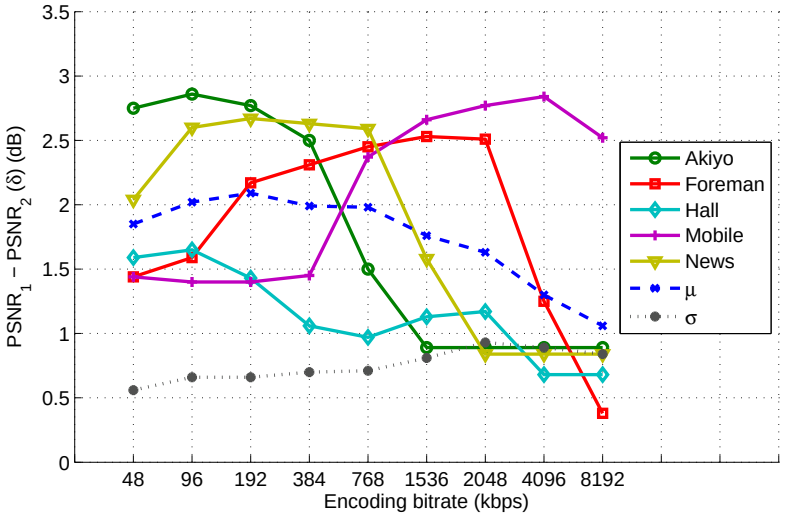


Fig. 6. δ values for test sequences.

(a) Raw video (b) 1536 kbps

(c) 384 kbps (d) 96 kbps

Fig. 7. Frame captures for the original and postmuxed video.

(a) 768 kbps (b) 192 kbps (c) 48 kbps

Fig. 8. Frame captures for the remuxed video. Please refer to Figure 7(a) to inspect the orginal frame.

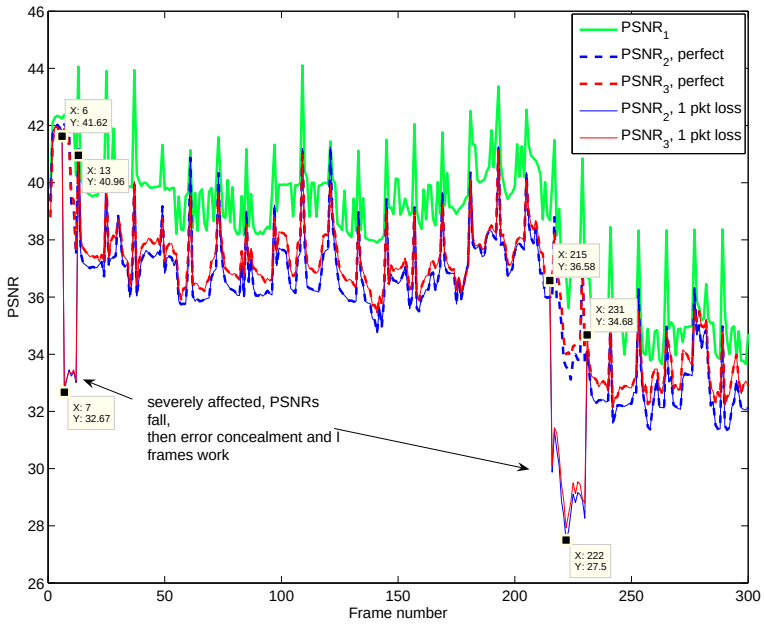


Fig. 9. PSNR performance for the entire sequence with encoding bitrate 1536 kbps. The raw video is *Foreman* sequence with 300 frames and CIF size. A single packet is dropped.

For error-free case, PSNR_1 and PSNR_2 are 36.1 and 36.51 dB, respectively. For the packet-loss case, these values fall down to 35.67 and 36.08 dB.

2) General case - Network Emulation: In Subsection IV-A1, the effect of minor channel degradation was studied. As the second scenario, we provide the experimental results obtained using RoSE testbed under packet erasure channel. The linux bridge in the system is configured to generate 10^{-1}, 10^{-2}, and 10^{-3} packet loss rates over the streaming traffic randomly.

The rate-distortion curves are shown in Figure 12. As expected, in general increasing packet loss incurs a larger PSNR penalty. The average PSNRs are larger for higher bitrates since streamed data are more interleaved and thus recovery is better. However, since the reconstructed frames are fewer than original, there is a coherency problem during evaluation, i.e. the corresponding frames are not compared all the time. For instance, there are 151 PSNRs for 8192k in Figure 10. This may be the reason why the system shows erratic behavior for varying bitrates.

For some bitrates, all the frames could be constructed. This is especially valid for PER = 10^{-3}. The number of

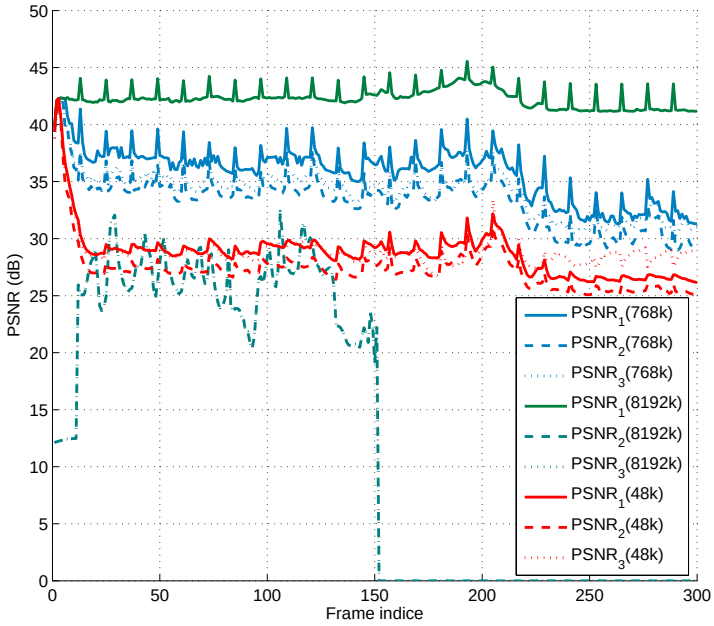


Fig. 10. PSNR performance for the entire sequence with encoding bitrates 96 kbps (red), 768 kbps (blue), and 8192 kbps (green). Solid lines are for $PSNR_1$, whereas dashed and dotted lines represent $PSNR_2$ and $PSNR_3$, respectively.The raw video is *Foreman* sequence with 300 frames and CIF size. PER = 10^{-3}

reconstructed frames are plotted in Figure 11. However, there is strange behavior where recovery at 10^{-1} is better than 10^{-2}. This pattern needs more investigation.

All this information is summarized in Table IV. We also employ offline post-processing by FFMPEG to replace missing frames with the adjacent available frames. This is a simple typical error concealment technique. More advanced reconstruction techniques such as averaging or macro-block level decomposition may also be used.

V. Conclusion

In this report, we have presented the second phase evaluation of MD schemes in terms of PSNR performance in the baseline and network-emulated scenarios. The first scenario entails the bare PSNR penalty due to decoding into a new codec, separating into multiple descriptions at Remux module and then merging them in the Postmux module. In other words, the cost of MDC employment in video transmission is studied. The transmission impairments are considered in the second scenario. As anticipated, the introduction of MDC incurs a PSNR penalty on the final performance. However, this penalty is relatively minor compared to the potential gains. For testbed evaluation, the distortion due to packet losses are observed. As future work, more MDC schemes will be implemented. Additionally, the performance will be evaluated for additional sequences against more diverse transmission simulations and real network environments.

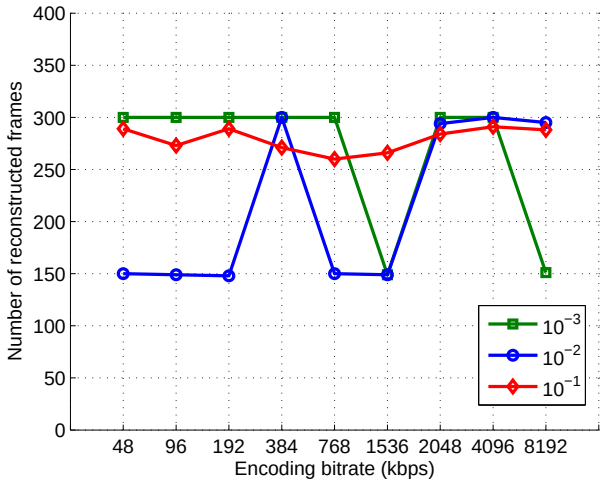


Fig. 11. Number of reconstructed frames at the Postmux for varying PER.

TABLE IV

AVERAGE PSNR VALUES FOR FOREMAN SEQUENCE ON THE TESTBED.

Packet loss	**PSNR type**	**48k**	**96k**	**192k**	**384k**	**768k**	**1536k**	**2048k**	**4096k**	**8192k**
	1	28.77	28.96	31.10	33.46	35.87	38.63	39.90	42.30	42.30
10^{-3}	2	27.34	39.82	28.93	31.13	33.42	12.11	36.87	40.13	12.59
	3	28.99	28.96	30.35	32.24	34.19	12.16	37.16	40.00	12.64
number of reconstructed frames		300	300	300	300	300	149	300	300	151
10^{-2}	2	11.75	12.07	11.46	30.72	11.76	11.66	31.41	36.59	33.77
	3	12.11	12.48	11.68	31.79	11.86	11.71	31.60	36.53	33.64
number of reconstructed frames		150	149	148	300	150	149	294	300	295
10^{-1}	2	22.80	17.88	24.86	16.59	13.57	14.26	17.55	17.77	17.36
	3	23.84	18.32	25.85	16.78	13.64	14.31	17.60	17.81	17.41
number of reconstructed frames		289	273	289	271	260	266	284	291	288

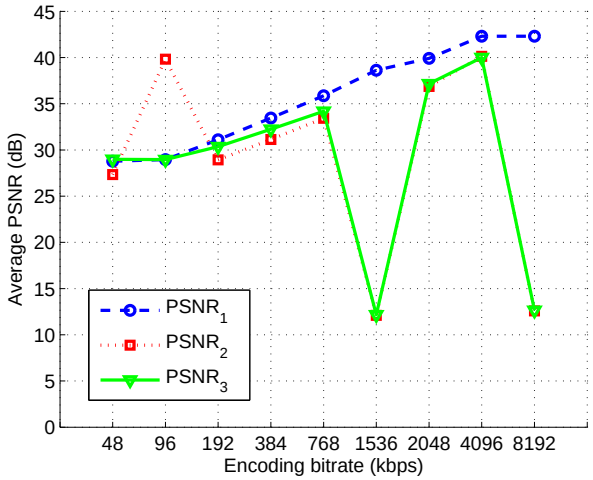
REFERENCES

[1] G. Gur and L. Abeni, “Bringing Multiple Description Coding and Scalable Video to Consumer Multimedia Communications,” submitted to *IEEE Communications Magazine*, 2009.

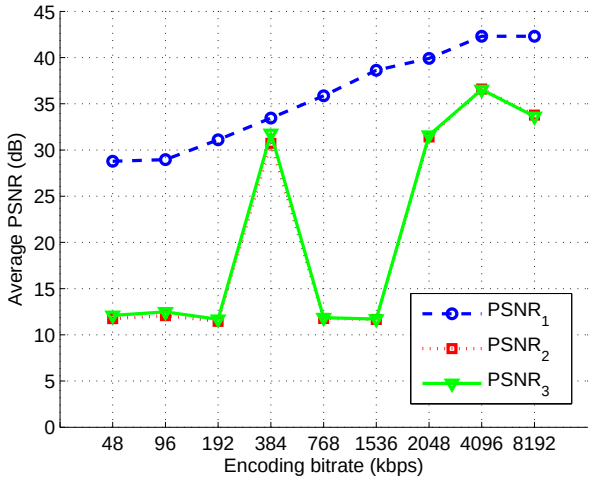
[2] V. K. Goyal, “Multiple description coding: Compression meets the network,” *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 74–93, September 2001.

[3] Y. Wang, A. Reibman, and S. Lin, “Multiple description coding for video delivery,” *Proceedings of the IEEE*, vol. 93, no. 1, pp. 57–70, Jan. 2005.

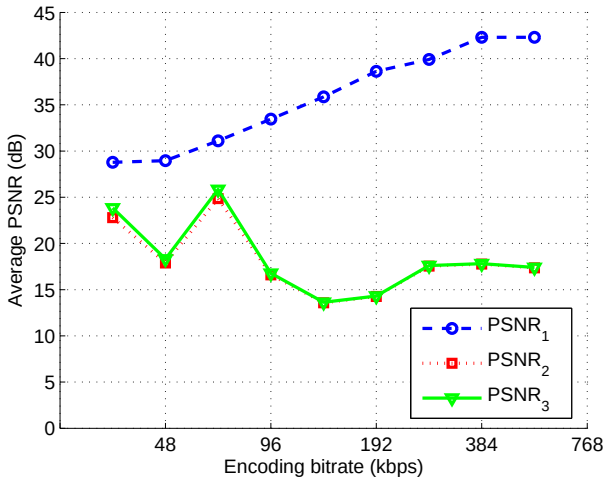
[4] A. Vitali and M. Fumagalli. (2005) Standard-compatible multiple-description coding (MDC) and layered coding (LC) of audio/video streams. Internet draft. [Online]. Available: http://tools.ietf.org/tools/rfcmarkup/rfcmarkup.cgi?draft=draft-vitali-ietf-avt-mdc-lc-00.txt



(a) PER=10^{-3}, random



(b) PER=10^{-2}, random



(c) PER=10^{-1}, random

Fig. 12. Rate-distortion curves at Postmux module for varying PER. Raw Foreman video (CIF, 300 frames) input and MPEG4-coded output at the Remux module.