

A parallel adaptive loop filter (JCTVC-B064)

Tomohiro Ikai
Yoshihiro Kitaura
Tomoyuki Yamamoto

SHARP Corporation

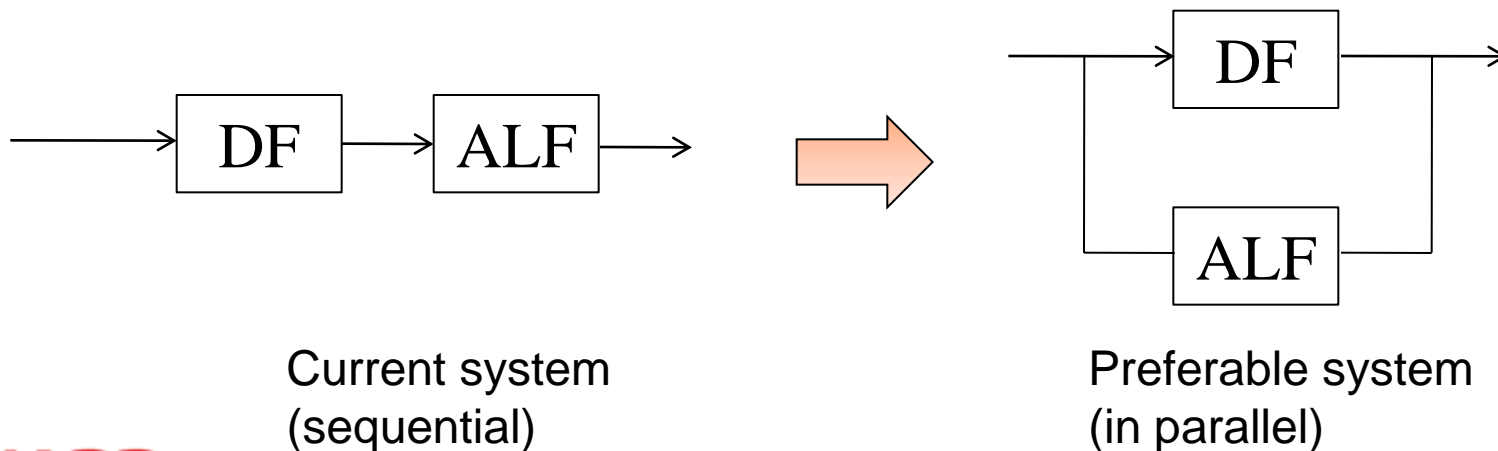
Contents

- Motivation
- Concept
- A parallel adaptive loop filter
- Experimental Results
- Conclusion

Motivation

- There are two efficient filter techniques
 - Deblocking filter (DF)
 - Adaptive Wiener loop filter (ALF)
- Problem
 - the filtering is time-consuming
 - shall be processed sequentially

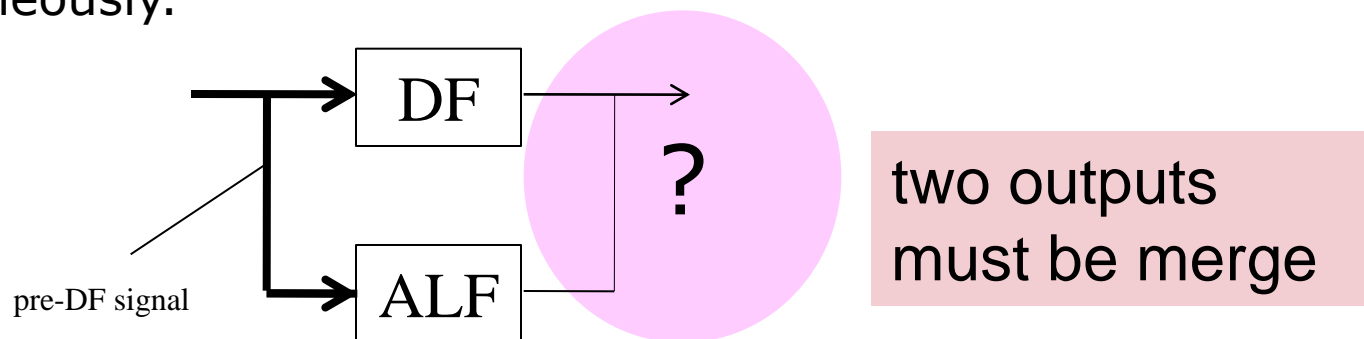
Can't we do DF and ALF in parallel?



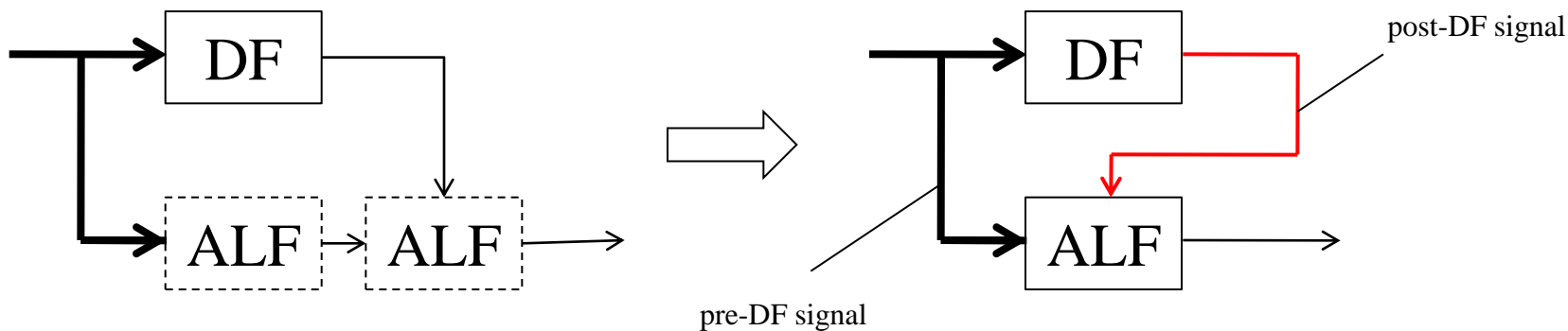
Concept

- Idea

- If the input of ALF is pre-DF signal, DF and ALF can be processed simultaneously.



- ALF can merge properly

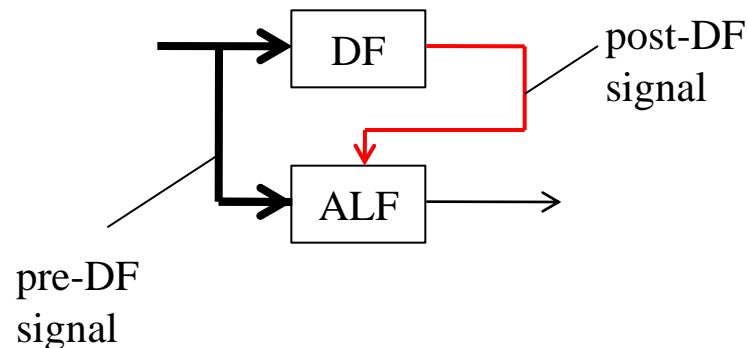


Two input system

A parallel adaptive loop filter

- Two input ALF system

- pre-DF signal
- post-DF signal

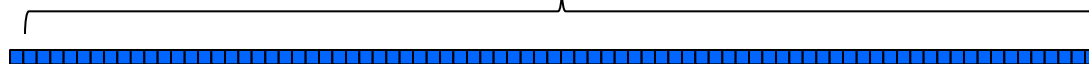


- Feature

- mainly use pre-DF signal
 - 3x3 to 9x9 pixels is used for pre-DF signal
 - 1 pixel is used for post-DF signal

$$s_{out} = \sum_{i=1}^N w_i \cdot s_i + w \cdot s_{post} + c$$

pre-DF signal is spatially filtered



max 81 pixels

post-DF signal is weighted

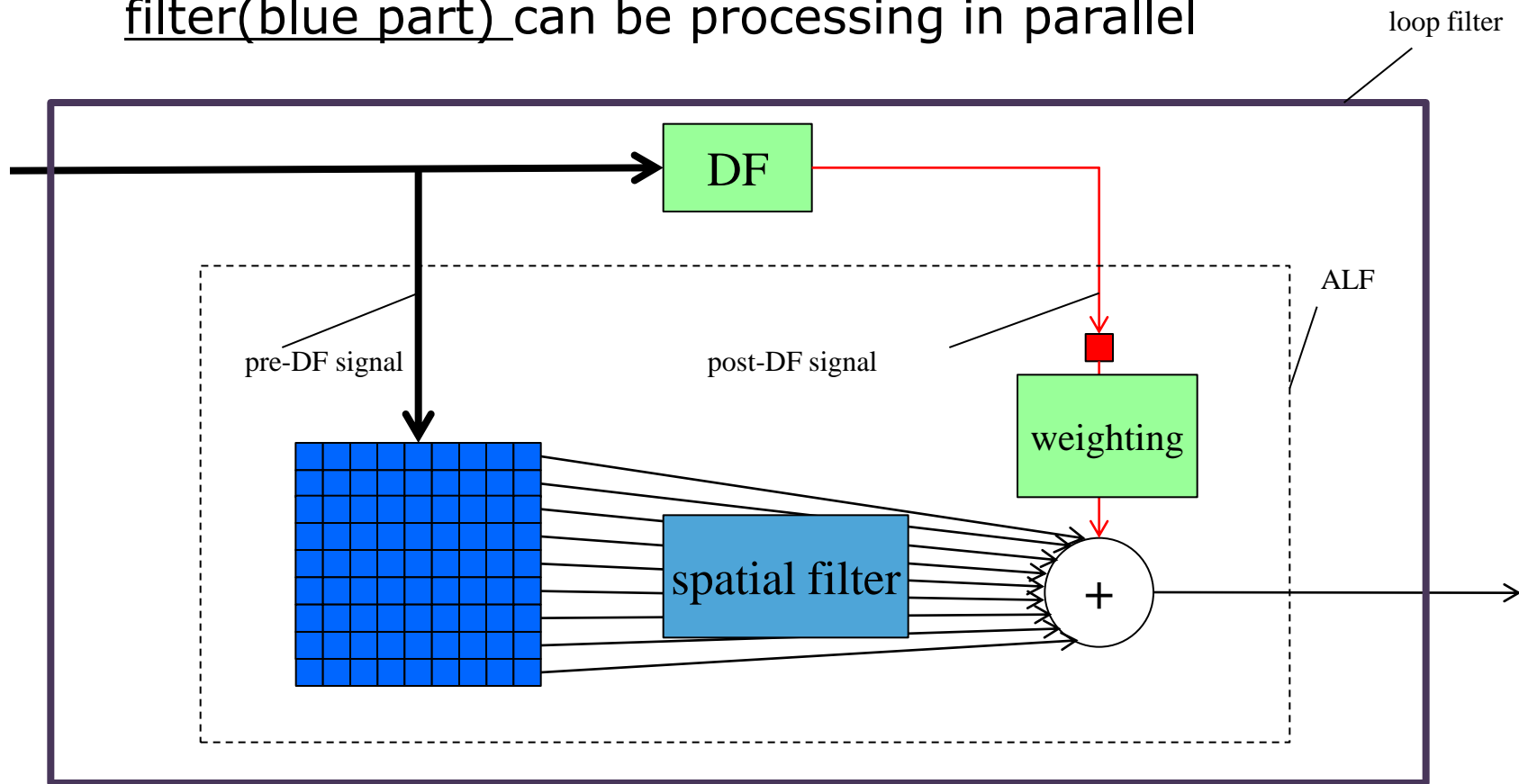


1 pixel

A parallel adaptive loop filter

- Parallelism

- DF plus ALF-weighting(green part) and ALF-spatial filter(blue part) can be processing in parallel

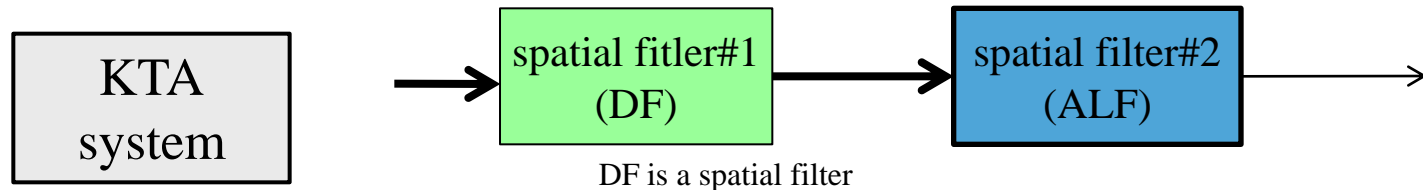


over 96% of ALF and DF can be processed in parallel

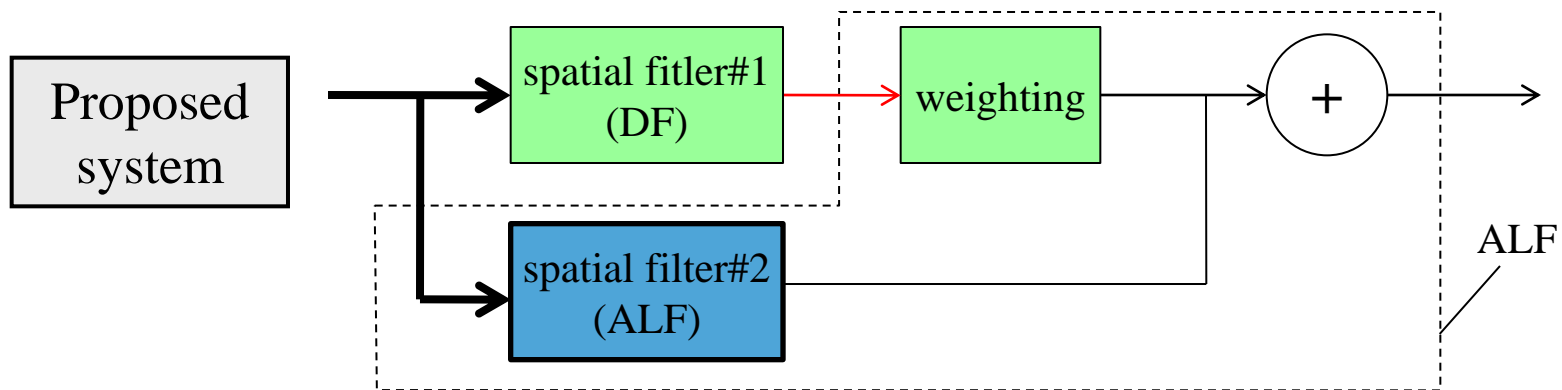
96% ($=25/(25+1)$) when $N=25$
98% ($=49/(49+1)$) when $N=49$
99% ($=81/(81+1)$) when $N=81$

A parallel adaptive loop filter

- Comparison with KTA system



two spatial filters are sequential



two spatial filters are parallel

Experiment results

- Implemented into KTA2.6r1
- Anchor: KTA2.6r1
- Test condition
 - In-loop filtering ad-hoc group condition
 - Conditions
 - CS1(HierB) and CS2(IPPP)
 - QPs
 - 26, 30, 34, 38
 - Tools
 - ExtMB, MDDT, MVComp, SIFO

Coding Efficiency (CS1, HierB)

- 1.55% bitrate reduction(0.057 dB)

Resolution	Sequence	NumFrames	Δ Bitrate(%)	Δ PSNR(dB)
A	Traffic	65	3.09	0.119
	People on Street	65	2.77	0.131
B	Kimono	49	0.93	0.033
	ParkScene	49	2.6	0.086
	Cactus	97	1.14	0.033
	BasketballDrive	97	2.61	0.074
	BQTerrace	129	1.95	0.037
C	BasketballDrill	97	2.11	0.084
	BQMall	129	0.61	0.027
	PartyScene	97	0.82	0.035
	RaceHorses	65	0.38	0.015
D	BasketballPass	97	1.8	0.082
	BQSquare	129	1	0.042
	BlowingBubbles	97	0.89	0.035
	RaceHorses	65	0.46	0.022
	Average (all)	-	1.55	0.057

Coding Efficiency (CS2, IPPP)

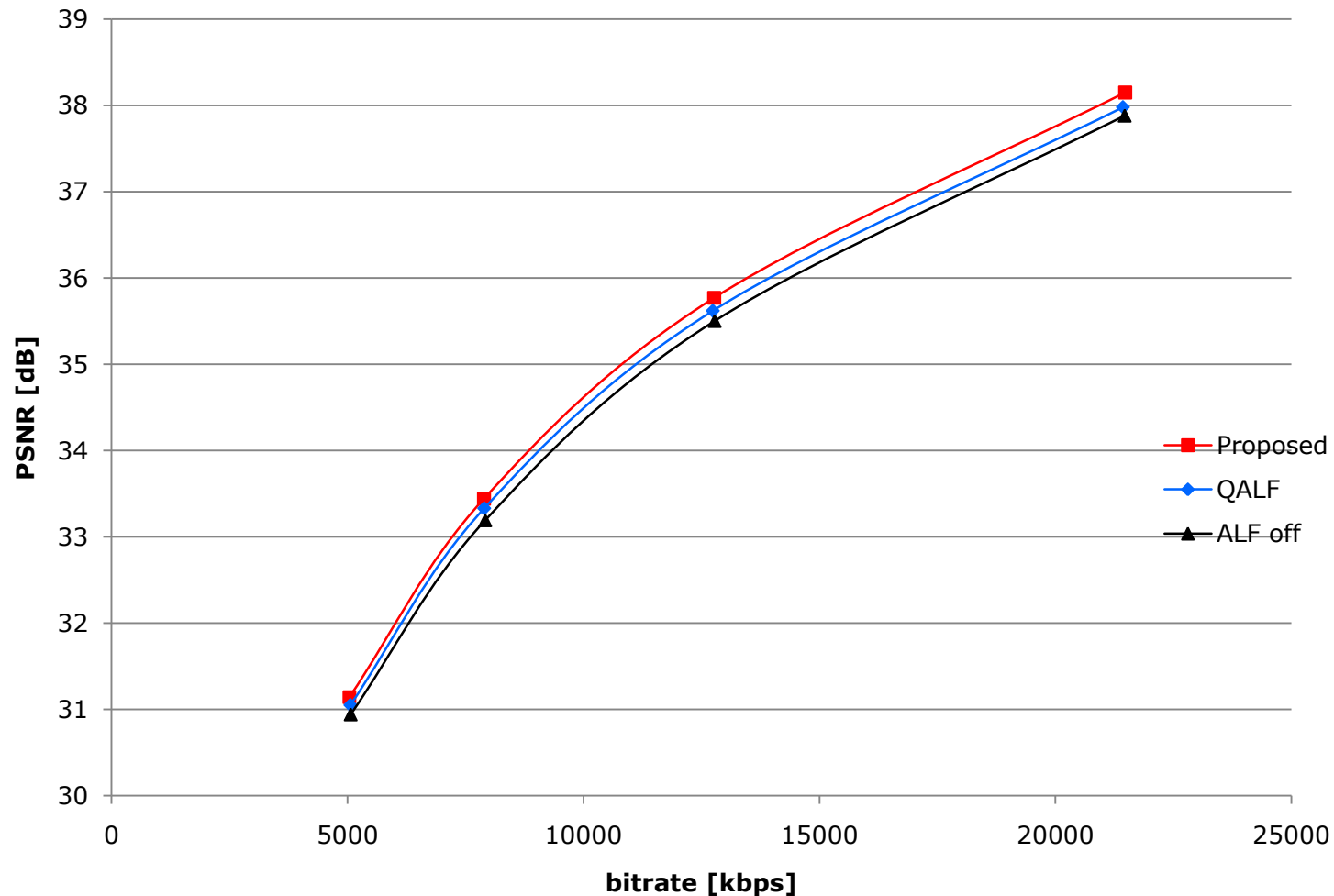
- 0.84% bitrate reduction(0.029 dB)

Resolution	Sequence	NumFrames	Δ Bitrate(%)	Δ PSNR(dB)
A	Traffic	65	1.37	0.048
	People on Street	65	0.94	0.044
B	Kimono	49	0.28	0.01
	ParkScene	49	1.55	0.048
	Cactus	97	0.76	0.022
	BasketballDrive	97	2.03	0.056
	BQTerrace	129	1.64	0.032
	BasketballDrill	97	0.53	0.019
C	BQMall	129	0.38	0.016
	PartyScene	97	-0.19	-0.005
	RaceHorses	65	0.13	0.006
	BasketballPass	97	1.27	0.06
D	BQSquare	129	0.03	0.002
	BlowingBubbles	97	0.5	0.019
	RaceHorses	65	0.52	0.024
	Vidyo1	129	1.19	0.041
E	Vidyo3	129	1.47	0.055
	Vidyo4	129	0.74	0.023
	Average (all)	-	0.84	0.029

RD performance

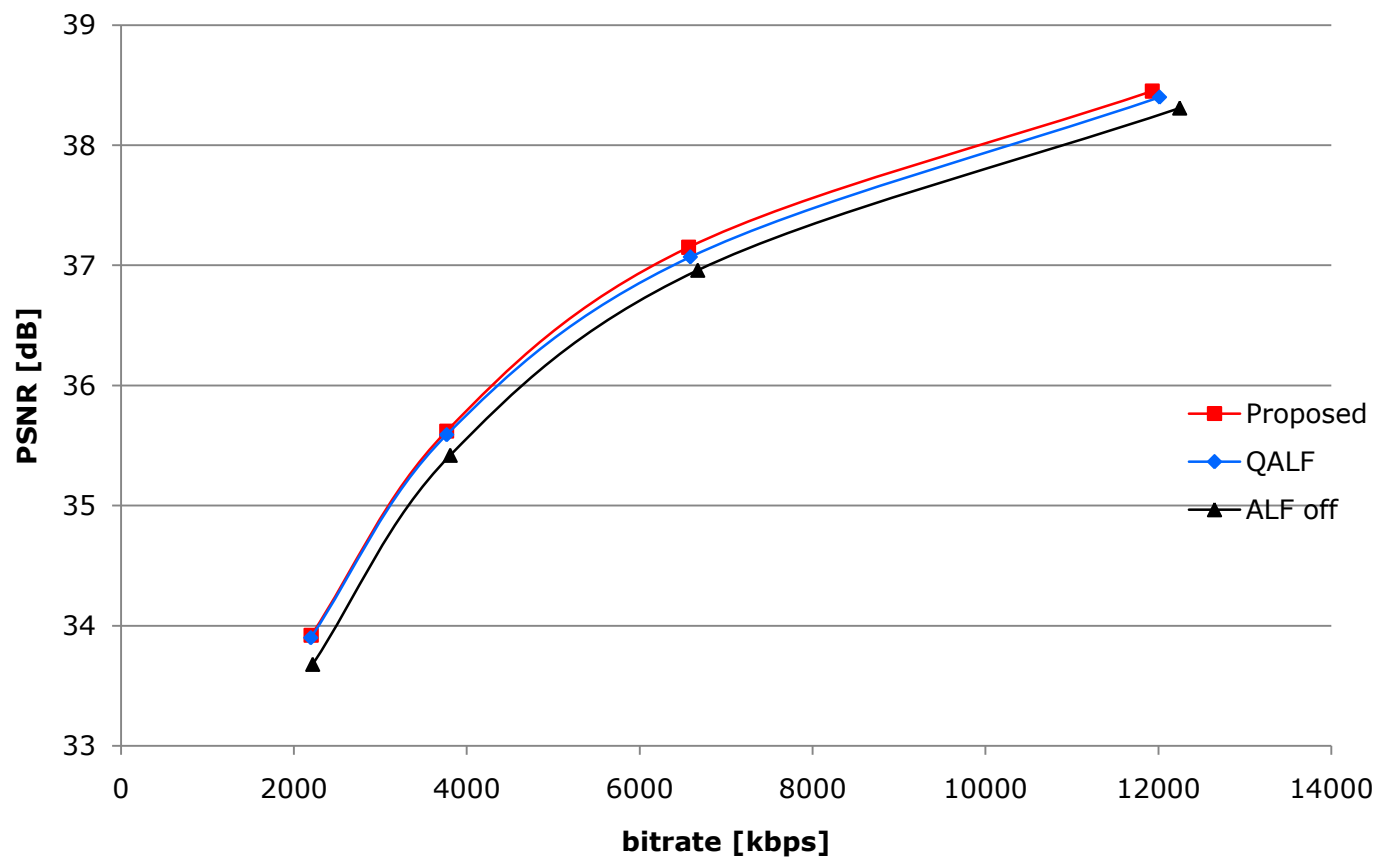
■ PeopleOnStreet(CS1)

- $\Delta\text{bdrate} = 2.77$ [%], $\Delta\text{bdsnr} = 0.131$ [dB]



RD performance

- BasketballDrive(CS2)
 - $\Delta b_{\text{rate}} = 2.03$ [%], $\Delta b_{\text{dsnr}} = 0.056$ [dB]



Conclusion

- A parallel adaptive loop filter is proposed
 - two inputs (pre-DF and post-DF)
 - pre-DF pixels are mainly used.
 - one post-DF pixel is used.
- Parallelism
 - over 96% of ALF and DF can be processed in parallel
- Coding efficiency
 - improve over QALF
 - 1.55 % bitrate reduction(0.057 dB) in CS1
 - 0.84 % bitrate reduction(0.029 dB) in CS2

propose to create in-loop filtering TE
consider parallel processing capability in the TE