



# Non-CE8.c.6: Multi-source SAO and ALF virtual boundary processing with cross9x9

C.-Y. Chen, C.-M. Fu, C.-Y. Tsai, Y.-W. Huang, S. Lei (MediaTek)  
S. Esenlik, M. Narroschke, T. Wedi (Panasonic)  
I. S. Chong, M. Karczewicz (Qualcomm)



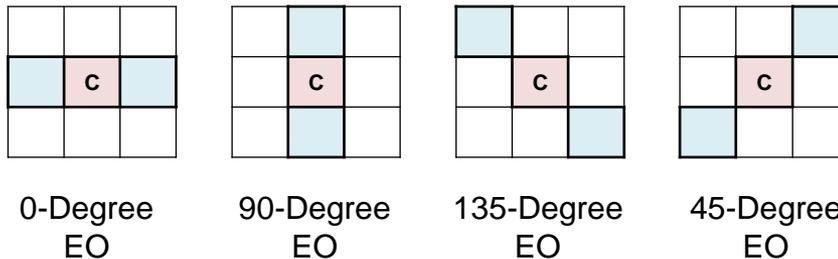
Presented by Yu-Wen Huang  
7<sup>th</sup> JCT-VC Meeting in Geneva  
21-30 November, 2011

# Overall Summary

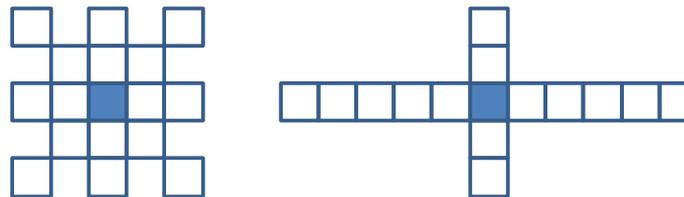
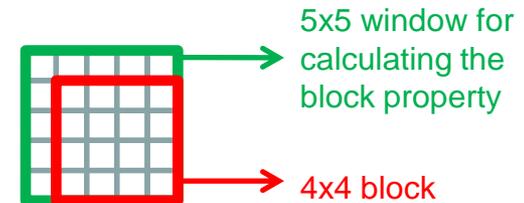
- In HM-4.0
  - SAO: 0.2 luma line, 0.2 chroma line
  - ALF: 4.1 luma lines, 4 chroma lines
- In the proposed virtual boundary (VB) processing
  - Luma VB: 3 pixels above the horizontal LCU boundary
  - Chroma VB: 1 pixels above the horizontal LCU boundary
  - Processing a pixel on one side of a VB does not use any pixel from the other side of the VB unless it is already in the buffer.
  - Change SAO and ALF input pixels (multi-source)
  - All SAO and ALF line buffers can be saved
  - 0-0.5% bit rate reduction
  - No encoding time change
  - 2-4% decoding time increase
  - The same visual quality

# Background Information

- Deblocking filter (DF)
  - Luma: read 4 pixels and write 3 pixels
  - Chroma: read 2 pixels and write 1 pixel
- Sample adaptive offset (SAO)
  - Edge offset (EO): need a 3x3 window for pixel classification

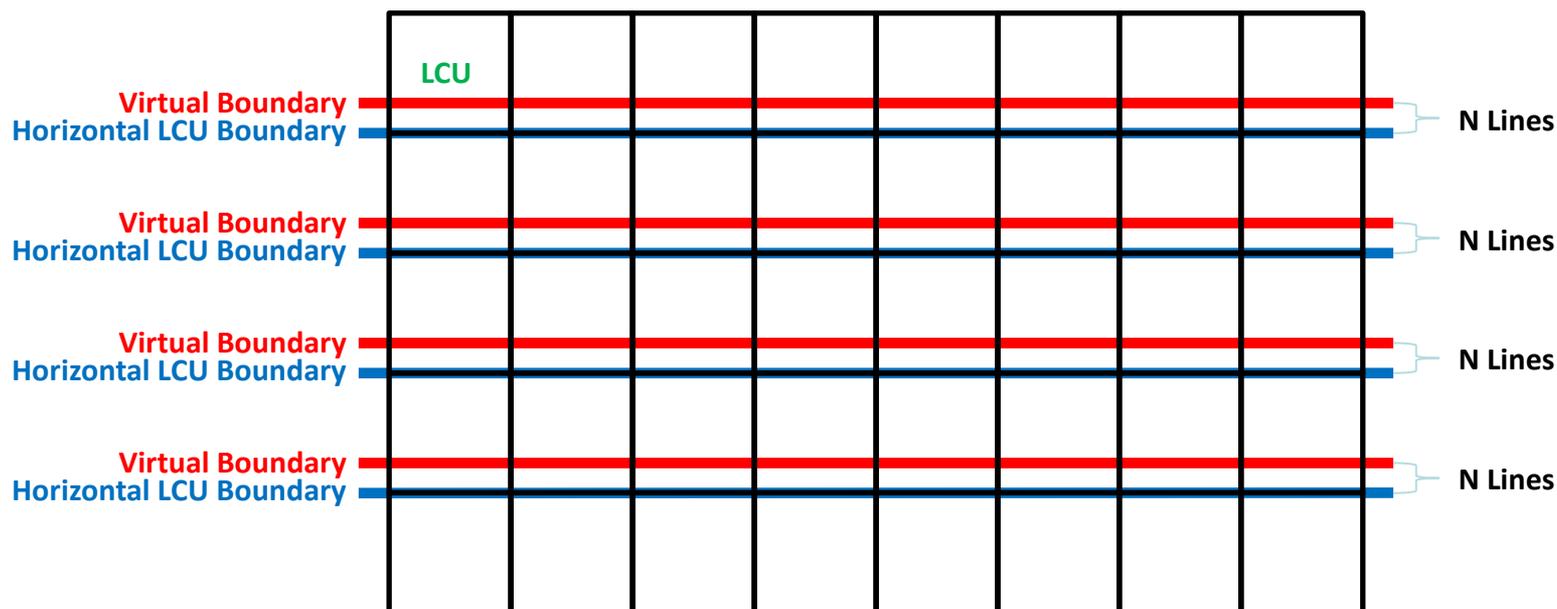


- Adaptive loop filter (ALF)
  - Calculate 4x4 block property for luma
  - Snowflake5x5
  - Cross11x5



# Virtual Boundary (VB)

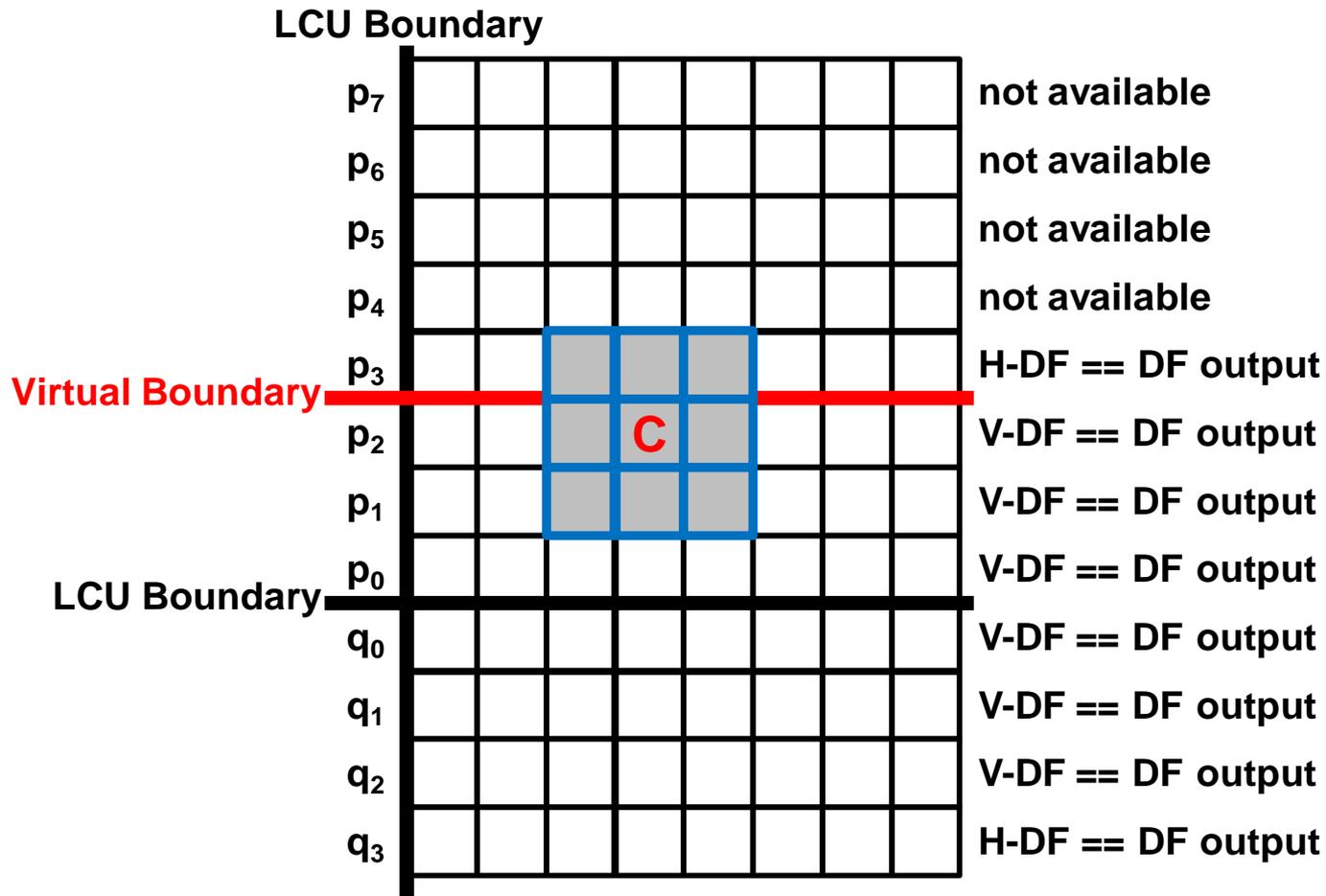
- $N=3$  for luma,  $N=1$  for chroma
- Processing a pixel on one side of a VB does not use any pixel on the other side of the VB unless it is already in the buffer.





# SAO VB Processing for Below-VB Pixels

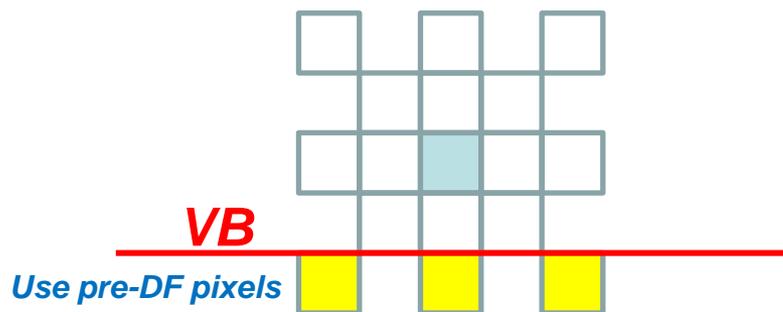
- Same as HM-4.0



# ALF VB Processing for Above-VB Pixels

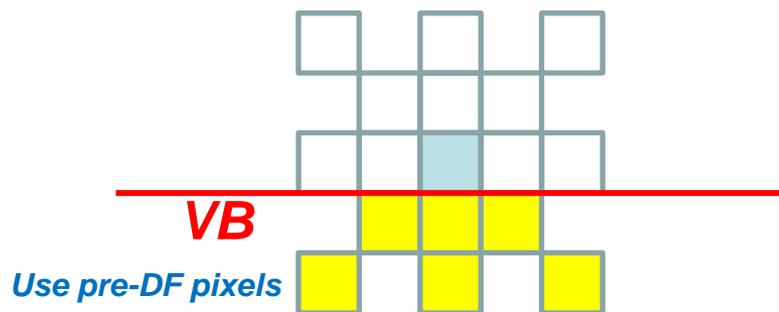
Snowflake5x5

2<sup>nd</sup> line: multi-source



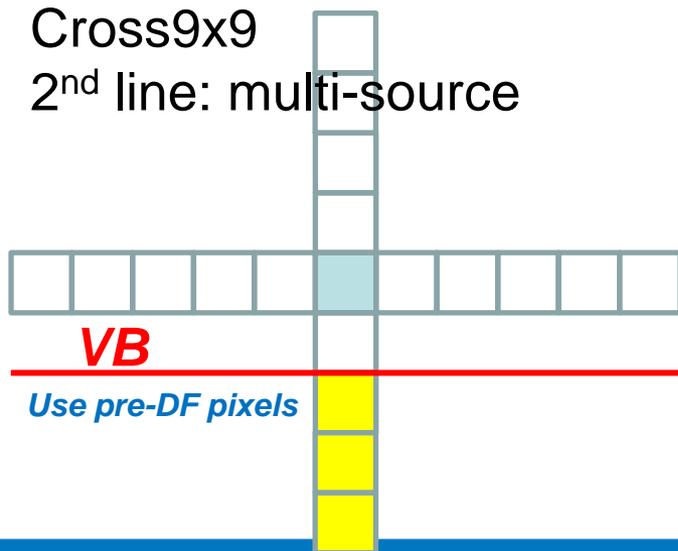
Snowflake5x5

1<sup>st</sup> line: multi-source



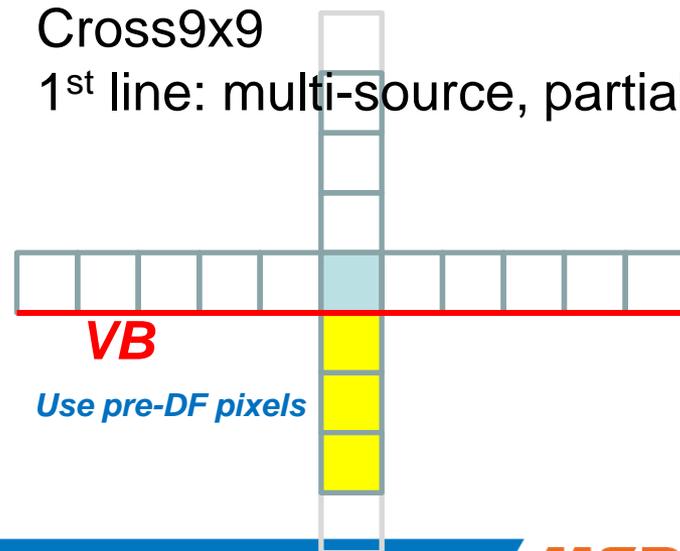
Cross9x9

2<sup>nd</sup> line: multi-source



Cross9x9

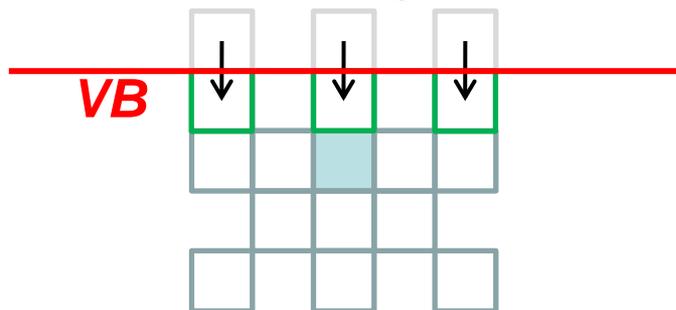
1<sup>st</sup> line: multi-source, partial filtering



# ALF VB Processing for Below-VB Pixels

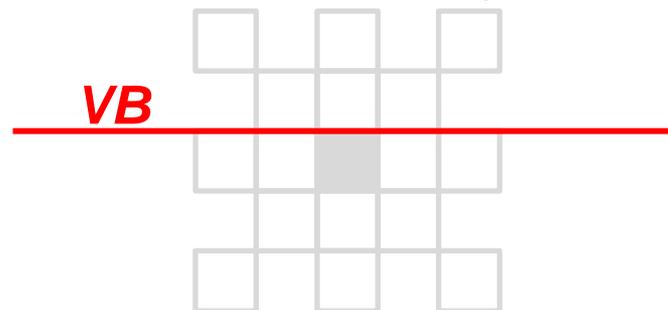
Snowflake5x5

2<sup>nd</sup> line: padding and averaging



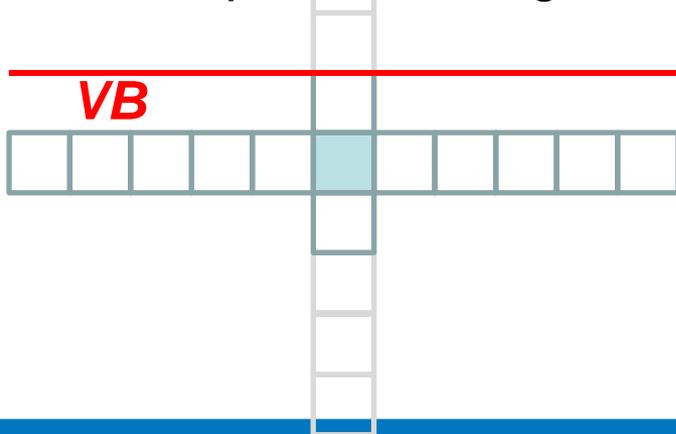
Snowflake5x5

1<sup>st</sup> line: skip filtering



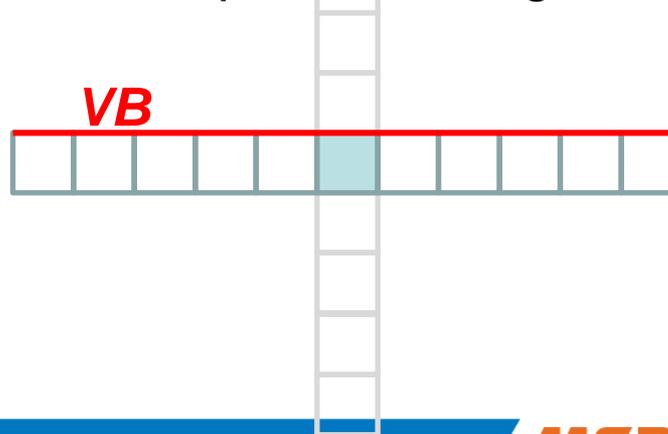
Cross9x9

2<sup>nd</sup> line: partial filtering



Cross9x9

1<sup>st</sup> line: partial filtering



# Simulation Results

- Anchor: JCTVC-F900
- 0-0.5% gain
- Similar encoding time
- 2-4% decoding time increase
- Same visual quality

	All Intra HE			All Intra LC		
	Y	U	V	Y	U	V
Class A	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Class B	0.0%	0.1%	0.1%	0.0%	0.0%	0.1%
Class C	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%
Class D	-0.2%	0.1%	0.1%	0.0%	0.0%	0.0%
Class E	-0.1%	0.2%	0.2%	0.0%	0.1%	0.1%
<b>Overall</b>	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%
	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%
Enc Time[%]	99%			100%		
Dec Time[%]	102%			101%		

	Random Access HE			Random Access LC		
	Y	U	V	Y	U	V
Class A	-0.1%	0.1%	-0.1%	0.0%	0.1%	0.0%
Class B	-0.1%	0.0%	0.1%	0.0%	0.0%	0.1%
Class C	-0.2%	-0.1%	0.0%	0.0%	0.0%	0.0%
Class D	-0.7%	-0.2%	-0.1%	0.0%	-0.2%	0.0%
Class E						
<b>Overall</b>	-0.2%	-0.1%	0.0%	0.0%	0.0%	0.0%
	-0.3%	-0.1%	0.0%	0.0%	0.0%	0.0%
Enc Time[%]	100%			100%		
Dec Time[%]	102%			102%		

	Low Delay B HE			Low Delay B LC		
	Y	U	V	Y	U	V
Class A						
Class B	-0.4%	-0.3%	0.1%	0.0%	0.2%	-0.1%
Class C	-0.4%	-0.1%	-0.1%	0.0%	0.0%	-0.1%
Class D	-0.9%	-0.1%	-0.5%	0.0%	0.3%	-0.1%
Class E	-0.5%	0.0%	0.8%	-0.2%	0.2%	-0.2%
<b>Overall</b>	-0.5%	-0.2%	0.0%	0.0%	0.2%	-0.1%
	-0.5%	-0.2%	0.0%	0.0%	0.1%	-0.1%
Enc Time[%]	101%			104%		
Dec Time[%]	104%			106%		

	Low Delay P HE			Low Delay P LC		
	Y	U	V	Y	U	V
Class A						
Class B	-0.2%	-0.1%	0.1%	0.0%	0.2%	0.0%
Class C	-0.1%	0.1%	-0.1%	0.0%	0.1%	-0.1%
Class D	-0.1%	-0.1%	0.0%	0.0%	0.2%	0.0%
Class E	-0.5%	0.5%	-0.4%	0.0%	0.1%	-0.2%
<b>Overall</b>	-0.2%	0.0%	-0.1%	0.0%	0.1%	-0.1%
	-0.2%	0.0%	-0.1%	0.0%	0.1%	-0.1%
Enc Time[%]	100%			98%		
Dec Time[%]	104%			99%		

# Conclusion

- Combine multi-source SAO and ALF VB processing and snowflake5x5+cross9x9
- Remove all SAO and ALF line buffers
- 0-0.5% coding efficiency gain
- Same visual quality