



REDEFINING MOBILITY



# JCT3V-C0054: Simplifications for adaptive luminance compensation in 3D-AVC

Jewon Kang, Ying Chen, Li Zhang, and Marta Karczewicz

# Introduction

- Adaptive luminance compensation (ALC) in 3D-AVC
  - Compensating the luminance discrepancy in inter-view pictures, and achieving around 2% BD-rate saving in 3D-AVC.
  - Using a set of prediction weights derived with the above and left regions of the current block and the corresponding block in a reference picture (See the figure [1] below)



[1] M. Mishurovskiy, A. Fartukov, I. Kovliga, J. Lee, "3D-CE2.a results on inter-view coding with adaptive luminance compensation," JCT3V-B0031

# Throughput Problem in Decoding

- Sequential 4x4 block-level process for prediction weight derivation in a MB partition (when 4x4 DCT transform is used).
  - In decoding, all the 4x4 blocks in an MB partition may undergo motion compensation and inverse T/Q in series even though they have the same motions in the partition.
- Inter-block dependency in a MB partition
  - The current block in a MB partition cannot be decoded until the above and the left block become available.

The current block to be coded:   
The decoded block: 

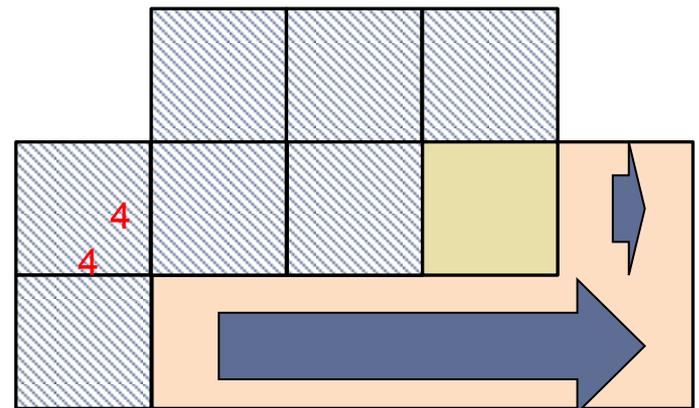
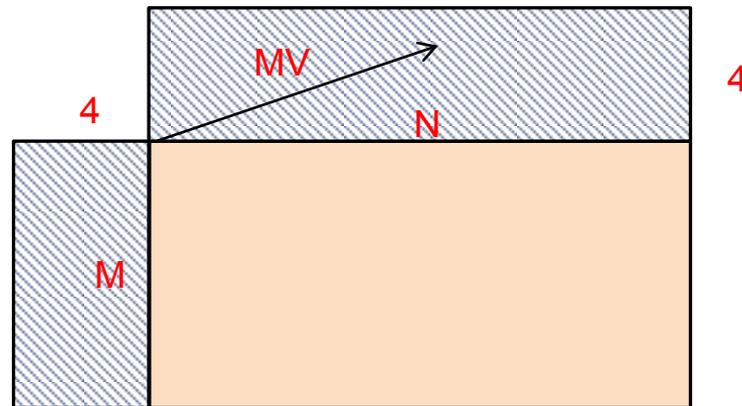


Figure: the ALC applied to a 16x8 MB partition when the transform size is 4

# Proposed Method

- One time prediction weight derivation per MB partition (when the partition is larger than or equal to 8x8 with no further sub-MB partitions)
  - No more inter-block dependency here
  - Applied to all luma pixels in the MB partition



# Proposed Method

- No more transform size dependency to decide M and N
  - The parameters are mainly decided by the MB partition (and the sub-MB partition)

mb_type	transform_size_8x8_flag equal to 1		transform_size_8x8_flag equal to 0	
	M	N	M	N
P_L0_16x16	16	16	8	8
P_L0_L0_16x8	16	8	4	4
P_L0_L0_8x16	8	16	4	4
P_8x8,_8x8ref0	8	8	4	4
P_Skip	Undefined		16	16

(a)

mb_type	M	N	mb_type (sub_mb_type)		M	N
P_L0_16x16	16	16	P_8x8, P_8x8ref0	P_L0_8x8	8	8
P_L0_L0_16x8	16	8		P_L0_8x4	4	4
P_L0_L0_8x16	8	16		P_L0_4x8	4	4
P_Skip	16	16		P_L0_4x4	4	4

(b)

Table1. Parameters (M and N) specifying the left and the top regions for prediction weights derivation: (a) the current design and (b) the proposed.

# Experimental Results

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)		Complexity estimate (ratio to anchor)		
	dBR, %	dPSNR, dB	dBR, %	dPSNR, dB	dBR, %	dPSNR, dB	dBR, %	dPSNR, dB	Encoder Time, %	Decoder Time, %	Rendering Time, %
S01	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	96.32	95.69	108.30
S02	0.02	0.00	0.00	0.00	0.02	0.00	0.00	0.00	99.92	98.82	106.58
S03	0.01	0.00	0.00	0.00	0.01	0.00	0.01	0.00	97.20	100.42	99.96
S04	-0.02	0.00	0.00	0.00	-0.01	0.00	-0.02	0.00	100.45	98.46	100.17
S05	0.08	0.00	0.00	0.00	0.07	0.00	0.02	0.00	100.68	94.68	91.50
S06	0.09	0.00	0.00	0.00	0.09	0.00	0.04	0.00	96.22	96.31	103.86
S08	0.09	0.00	0.00	0.00	0.09	0.00	0.04	0.00	97.14	98.52	96.92
Average	0.04	0.00	0.00	0.00	0.04	0.00	0.02	0.00	98.27	97.56	101.04

Proposed algorithm VS ATM6.0 rev1.

See Appendix to know how to measure the decoding execution time (suggested by cross-checker)

# Conclusion

## ■ Proposed Method

- Simplify the ALC in 3D-AVC, so that the 4x4 block-level processing can be minimized (only allowed for some sub-MB partitions) while the design is still consistent with the original.
- Improvement in decoding throughput and only minor change in coding efficiency (~0.04 BD-rate increase).

# Thank you!

Special thank you Samsung for  
cross-checking (JCT3V-C0150)

# Appendix

- How to measure the decoding time (as suggested by the cross-checker) while avoiding possible fluctuation.
  - Job1: decoding of 7 reference bit-stream + decoding of 7 tested bit-stream : ( ref1,test1)
  - Job2: decoding of 7 tested bit-stream + decoding of 7 reference bit-stream : ( ref2,test2)
  - Two decoding times were evaluated:
    - (Test1 VS Ref2) and (Test2 VS Ref1)
  - Make them average to the decoding time.