



Sample Adaptive Offset with LCU-based Syntax

Chih-Ming Fu, Ching-Yeh Chen, Chia-Yang Tsai, Yu-Wen Huang, and Shawmin Lei
(MediaTek)

In Suk Chong and Marta Karczewicz
(Qualcomm)



Overall Summary

- LCU-based syntax for Sample Adaptive Offset (SAO)
 - Allow SAO type and offsets specified LCU by LCU, instead of regions
 - More flexible
- Potentially, more overhead to code
 - Use merge and run-length codes for better entropy coding
- This contribution reported the results for SAO on luma with the new LCU-based syntax
- Results:
 - Encoding and decoding times were similar to those of the anchor.
 - BD-rate degradations are small and as follows:

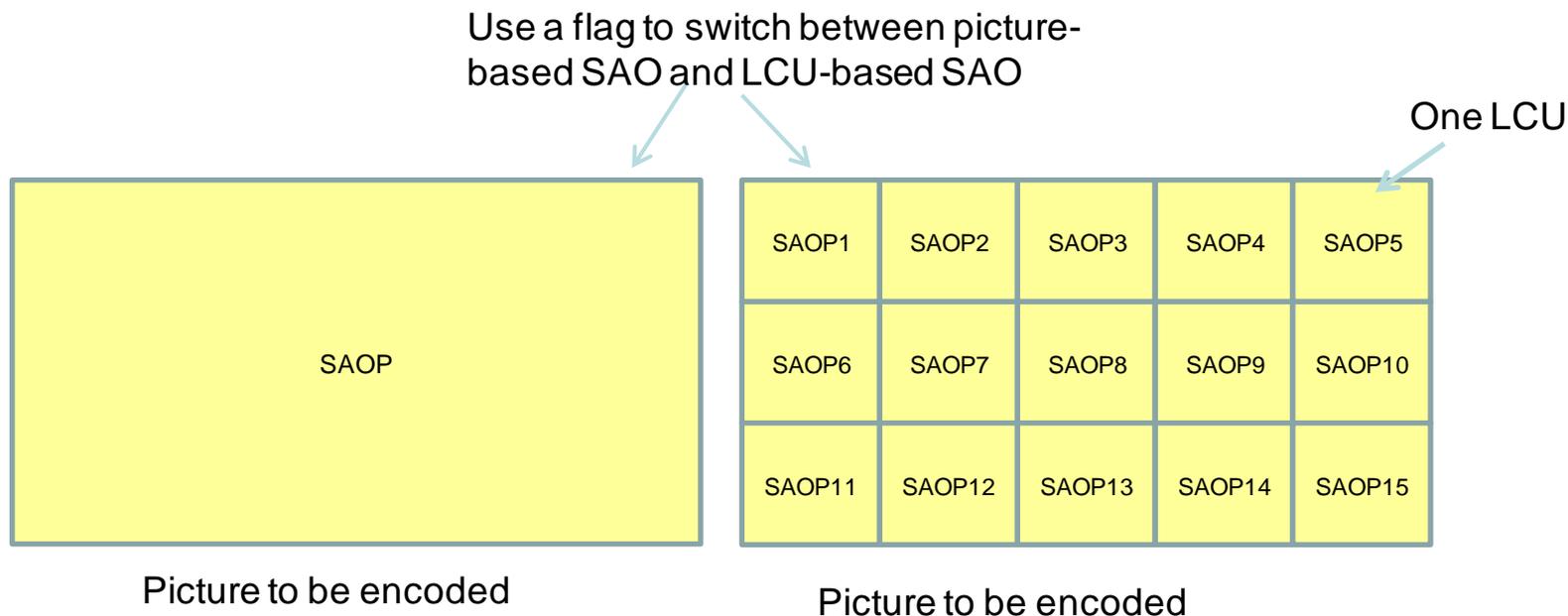
	HE-AI	HE-RA	HE-LD	LC-AI	LC-RA	LC-LD
BD-rate	0.0%	0.1%	0.1%	0.0%	0.0%	0.1%

Local Adaptation of SAO

- For each “region”:
 - First signal its type
 - one of 4 Band Offset (BO) types and 4 Edge Offset (EO) types
 - Then signal the offsets associated with the type
- In this proposal, the “region” would be LCU
 - More flexible
- Potential problem: More overhead to send
 - Need different syntax and entropy coding methods to efficiently send the overhead

Proposed Method – Adaptation Level

- A `sao_picture_level_flag` is used to signal picture or LCU level adaptation.
 - That signal the level that SAO parameter set can be changed.
- SAO parameter set (SAOP):
 - one SAO type and the offsets of the selected SAO type



Proposed Method – LCU Adaptation

- For the first row, run-length codes are used to reduce redundancy.
 - The LCUs in a run share the same SAO parameter set
 - Add BO offset prediction types (predicted from left LCU)
- For other rows, repeat, merge, and differential coding of runs are used.
 - `sao_repeat_row_flag`: Signal a repeat of the runs and SAO parameter sets of the previous row
 - `sao_merge_above_flag`: Signal that the same SAO parameter set of the above LCU is used for the current run
 - `sao_run_num_diff`: Differential coding of the current run length
 - Predictor is the residual run length of the above LCU
 - Add BO offset prediction types (predicted from upper LCU)

An Example of LCU Adaptation

use the SAOP of the above LCU

SAOP1 sao_run_num =2 no merge-above	SAOP1	SAOP1	SAOP2 sao_run_num =1 no merge-above	SAOP2
SAOP3 sao_run_num_diff =0 sao_merge_above_flag =0 sao_repeat_row_flag =0	SAOP3	SAOP3	SAOP2 sao_run_num_diff =0 sao_merge_above_flag =1	SAOP2
SAOP4 sao_run_num_diff =2 sao_merge_above_flag =0 sao_repeat_row_flag =0	SAOP4	SAOP4	SAOP4	SAOP4

A picture to be encoded

Simulation Result

- Anchor: JCTVC-E700
- Software platform:
HM-3.0
- Cross-check report:
JCTVC-F439
 - Thank Elena Alshina
(Samsung)

	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	0.0	0.0	0.0	0.0	0.0
Class C	0.0	0.0	0.0	0.0	0.0	0.0
Class D	0.0	0.0	0.0	0.0	0.0	0.0
Class E	0.0	0.0	0.0	0.0	0.0	0.0
Overall	0.0	0.0	0.0	0.0	0.0	0.0
Enc Time[%]	100%			99%		
Dec Time[%]	100%			100%		

	Random Access HE			Random Access LC		
	Y	U	V	Y	U	V
Class A	0.0	-0.2	0.0	0.0	0.0	0.3
Class B	0.1	0.0	0.1	0.0	0.0	0.1
Class C	0.1	0.1	0.1	0.0	0.0	0.0
Class D	0.1	0.0	-0.1	0.0	0.0	0.1
Class E						
Overall	0.1	0.0	0.0	0.0	0.0	0.1
Enc Time[%]	101%			100%		
Dec Time[%]	101%			100%		

	Low delay B HE			Low delay B LC		
	Y	U	V	Y	U	V
Class A						
Class B	0.1	0.2	0.2	0.1	0.1	-0.1
Class C	0.0	0.1	0.1	0.1	0.1	-0.1
Class D	0.0	0.0	0.1	0.0	-0.2	-0.2
Class E	0.4	0.1	0.1	0.5	0.2	-0.1
Overall	0.1	0.1	0.1	0.1	0.1	-0.1
Enc Time[%]	100%			100%		
Dec Time[%]	100%			101%		

Conclusions

- Sample adaptive offset with LCU-based Syntax
- Allowed SAO parameters to be adaptively changed at any largest coding unit (LCU)
- Encoding and decoding times were almost unchanged.
- Propose to adopt SAO with LCU-based syntax into HM

	HE-AI	HE-RA	HE-LD	LC-AI	LC-RA	LC-LD
BD-rate	0.0%	0.1%	0.1%	0.0%	0.0%	0.1%