



JCTVC-T0121 ON LOSSLESS CODING

Xiaoyu Xiu, Yan Ye, Yuwen He
InterDigital Communications, Inc.
Feb. 2015

INTERDIGITAL.

Creating the Living Network

Introduction

- In the current design, lossless coding is only enabled at CU level
 - Set `transquant_bypass_enabled_flag` to 1 in PPS
 - Set `cu_transquant_bypass_flag` to 1 at CU level → inverse quantization, inverse transform and in-loop filtering are bypassed
 - When `transquant_bypass_enabled_flag` is 0, `cu_transquant_bypass_flag` is not signaled and inferred to be 0
- In this contribution, two **independent** lossless coding related modifications are proposed:
 - Enable picture/slice level lossless signaling for typical lossless applications
 - Simplified transform quadtree splitting for lossless coded CUs
- Motivations:
 - Simplify syntax parsing
 - Allow to optimize decoder initialization
 - Encoding and decoding speed-up

Lossless signaling Option #1: PPS extension

pps_scc_extensions() {	Descriptor
if(!transquant_bypass_enabled_flag)	
transquant_bypass_default_flag	
residual_adaptive_colour_transform_enabled_flag	u(1)
}	

transquant_bypass_default_flag specifies the inferred value of **cu_transquant_bypass_flag** when **transquant_bypass_enabled_flag** is equal to 0. When **transquant_bypass_default_flag** is not present, it is inferred to be equal to 0.

cu_transquant_bypass_flag equal to 1 specifies that the scaling and transform process as specified in subclause 8.6 and the in-loop filter process as specified in subclause 8.7 are bypassed. When **cu_transquant_bypass_flag** is not present, it is inferred to be equal to **transquant_bypass_default_flag**.

slice_segment_header() {	Descriptor
.....	
if(sample_adaptive_offset_enabled_flag && !transquant_bypass_enabled_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
.....	
if(!transquant_bypass_enabled_flag)	
slice_qp_delta	se(v)
.....	
}	

Lossless signaling Option #1: bitstream conformance requirements

- Bitstream conformance requirement for syntax elements related to transform, quantization and loop-filtering in PPS and PPS range extension
 - transform_skip_enabled_flag
 - cu_qp_delta_enabled_flag
 - pps_cb_qp_offset, pps_cr_qp_offset
 - pps_slice_chroma_qp_offsets_present_flag
 - loop_filter_across_tiles_enabled_flag
 - pps_loop_filter_across_slices_enabled_flag
 - deblocking_filter_override_enabled_flag
 - pps_scaling_list_data_present_flag
 - chroma_qp_offset_list_enabled_flag
 - log2_sao_offset_scale_luma, log2_sao_offset_scale_chroma
 - deblocking_filter_control_present_flag
 - pps_deblocking_filter_disabled_flag
 - pps_extension_present_flag, pps_scc_extensions_flag
- The above **syntax elements in red shall be set to 0** and the **syntax elements in green shall be set to 1** when transquant_bypass_default_flag is equal to 1

Lossless signaling Option 2: slice segment header

slice_segment_header() {	Descriptor
.....	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(!transquant_bypass_enabled_flag)	
transquant_bypass_default_flag	
.....	
if(sample_adaptive_offset_enabled_flag && !transquant_bypass_default_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
.....	
if(!transquant_bypass_default_flag)	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag && !transquant_bypass_default_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
}	
if(chroma_qp_offset_list_enabled_flag && !transquant_bypass_default_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag && !transquant_bypass_default_flag)	
deblocking_filter_override_flag	u(1)
.....	
}	

Lossless signaling Option 2: slice segment header

transquant_bypass_default_flag specifies the inferred value of **cu_transquant_bypass_flag** for the coding units in the current slice when **cu_transquant_bypass_flag** is not present. When **transquant_bypass_default_flag** is not present, it is inferred to be equal to 0.

cu_transquant_bypass_flag equal to 1 specifies that the scaling and transform process as specified in subclause 8.6 and the in-loop filter process as specified in subclause 8.7 are bypassed. When **cu_transquant_bypass_flag** is not present, it is inferred to be equal to **transquant_bypass_default_flag**.

Simplified Transform Quadtree Splitting (1)

- Transform quadtree splitting does not significantly impact the efficiency of lossless coding, due to skipped transform
- Bypassing transform quadtree splitting can significantly speed up encoding by skipping the RDO process of TU splitting.
- Proposed to conditionally bypass split_transform_flag signaling for lossless coded CUs
 - For all the intra-coded CUs, and all the inter-coded and IBC-coded CUs with block size from 64x64 to 32x32, split_transform_flag is not signaled and inferred to the default value.
 - For 8x8 and 16x16 CUs coded with inter mode or IBC mode, transform quadtree splitting is allowed.

transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	Descriptor
if((!cu_transquant_bypass_flag (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= 4)) &&	
log2TrafoSize <= MaxTbLog2SizeY &&	
log2TrafoSize > MinTbLog2SizeY &&	
trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth == 0)))	
split_transform_flag[x0][y0][trafoDepth]	ae(v)
.....	
}	

Simplified Transform Quadtree Splitting (2)

- Encoder-only implementation
 - For all the intra-coded CUs, and all the inter-coded and IBC-coded CUs with block size from 64x64 to 32x32, encoder only tests the transform quadtree partition indicated by the default value of `split_transform_flag`.
 - For 8x8 and 16x16 CUs coded with inter mode or IBC mode, encoder tests the cases with and without transform quadtree splitting.

Simulations

- The proposed lossless signaling Option 1 (PPS extension) was implemented on SCM-3.0
 - Option 2 has similar performance and encoding/decoding time as Option 1
- Three sets of simulations
 - **Setting one:** the proposed transform quadtree splitting is not applied
 - **Setting two:** the proposed transform quadtree splitting is applied
 - **Setting three:** the proposed transform quadtree splitting is implemented as encoder-only method

Thanks to Qualcomm for the cross-check!

Results of Setting One (lossless signaling only)

- Minimal impact on coding performance and encoding time
- The decoding time is reduced by 10%.

	All Intra				Random Access				Low Delay B			
	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)
RGB, text & graphics with motion, 1080p & 720p	0.0%	0.0%	-0.1%	0.0%	0.0%	-0.1%	-0.3%	0.0%	0.0%	-0.1%	-0.6%	0.0%
RGB, mixed content, 1440p & 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
RGB, Animation, 720p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
RGB, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, text & graphics with motion, 1080p & 720p	0.0%	0.0%	-0.1%	0.0%	0.0%	-0.1%	-0.3%	0.0%	0.0%	-0.2%	-0.5%	0.0%
YUV, mixed content, 1440p & 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, Animation, 720p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Enc Time[%]	102%				102%				103%			
Dec Time[%]	89%				91%				91%			

Results of Setting Two (lossless signaling + proposed TU splitting condition)

- Encoding time reduction: 15%
- Decoding time reduction: 10%.
- Minimal performance impact: BD rate increase of 0%, 0.1% and 0.1% for AI, RA and LB.

	All Intra				Random Access				Low Delay B			
	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)
RGB, text & graphics with motion, 1080p & 720p	0.0%	0.0%	-0.1%	0.1%	0.1%	0.1%	-0.2%	0.4%	0.1%	0.0%	-0.4%	0.5%
RGB, mixed content, 1440p & 1080p	0.0%	0.0%	0.0%	0.1%	0.1%	0.1%	0.0%	0.1%	0.1%	0.1%	0.0%	0.1%
RGB, Animation, 720p	0.0%	0.0%	0.0%	0.0%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%
RGB, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, text & graphics with motion, 1080p & 720p	0.1%	0.1%	-0.1%	0.2%	0.1%	0.0%	-0.1%	0.1%	0.0%	0.0%	-0.3%	0.1%
YUV, mixed content, 1440p & 1080p	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, Animation, 720p	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Enc Time[%]	80%				85%				89%			
Dec Time[%]	85%				94%				93%			

Results of Setting Three (lossless signaling + encoder only TU splitting condition)

- Encoding time reduction: 12%
- Decoding time reduction: 10%
- Minimal performance impact: BD-rate increase of 0.1%, 0.1% and 0.1% for AI, RA and LB.

	All Intra				Random Access				Low Delay B			
	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)	Bit-rate change (Total)	Bit-rate change (Average)	Bit-rate change (Min)	Bit-rate change (Max)
RGB, text & graphics with motion, 1080p & 720p	0.0%	0.0%	0.0%	0.1%	0.1%	0.1%	-0.1%	0.5%	0.1%	0.1%	-0.3%	0.5%
RGB, mixed content, 1440p & 1080p	0.1%	0.1%	0.0%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%
RGB, Animation, 720p	0.0%	0.0%	0.0%	0.0%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%
RGB, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, text & graphics with motion, 1080p & 720p	0.2%	0.1%	0.0%	0.3%	0.1%	0.1%	-0.1%	0.2%	0.1%	0.0%	-0.2%	0.2%
YUV, mixed content, 1440p & 1080p	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%
YUV, Animation, 720p	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
YUV, camera captured, 1080p	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Enc Time[%]	85%				89%				91%			
Dec Time[%]	85%				94%				95%			

Closing Remarks

- Typical lossless video applications use picture/slice level lossless coding
- Propose to two *independent* modifications for lossless coding
 - Add transquant_bypass_default_flag in PPS SCC extension or slice segment header
 - Simplify transform quadtree splitting for lossless coding
- Coding performance
 - High level lossless signaling: 10% decoding time reduction with small coding gain
 - Simplified transform quadtree splitting: 15% encoding time reduction, and little (0.1%) performance loss.
- Suggest to adopt into SCC