|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extension Development**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  2nd Meeting: Shanghai, CN, 13–19 Oct. 2012 | Document: JCT3V-B0134r4 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Results on Weighted Prediction in 3D Video Coding** | | |
| *Status:* | Input Document | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Jiwook Jung, Hongbin Liu  Jaewon Sung, Jie Jia, Sehoon Yea  221, Yangjae-Dong, Seocho-Gu, Seoul 137-130, Korea | Tel: Email: | +82-2-6912-6477 [jiwook.jung@lge.com](mailto:jiwook.jung@lge.com) |
| *Source:* | LG Electronics | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

In this contribution inheritance of weighted prediction (WP) parameters for a dependent view from the base view is proposed. A flag indicating whether the WP parameters on the dependent view will be inherited from the base view is introduced in the slice header. It is shown that inheritance of weighted prediction parameters further improves coding efficiency of dependent views for 3D fade sequences generated in the same manner as was done in the HEVC standardization process. The proposed algorithm reportedly showed

# Introduction

Weighted Prediction (WP) is known for its benefits in coding fading in/out sequences [1][2] and part of the HEVC specification. In 3D video such as stereoscopic and multi-view videos, the fading in/out effects happen in all views simultaneously with the almost same magnitude. This contribution proposes inheriting the WP parameters for the dependent views from the base view. Because the WP parameters for inter-view reference picture of the dependent view cannot be inherited from base view, they are set to default values such that 1.0 for weight and 0.0 for offset. A flag indicating whether WP parameters on the dependent view will be inherited from the independent view is introduced in the slice header.

# Proposed Algorithm

If the flag, **base\_pred\_weight\_table\_flag,** in slice header of a dependent view is equals to 1, The WP parameters for the dependent view are inherited from the base view. Table 1 summarizes the pseudo code of the inheritance process. In case of the inter-view reference picture of the dependent view, the WP parameters are set to default values in the WP parameter reset process; weight=1.0, offset=0.0.

Table 1 pseudo code of WP parameter inheritance

|  |
| --- |
| if (base\_pred\_weight\_table\_flag)  {  reset WP parameter table of current dependent view  for( X=0 ; X< (B\_SLICE?2:1) ; X++ )  {  for (i = 0; i<=num\_ref\_idx\_lX\_active\_minus1; i++)  {  for (j = 0; j<base\_num\_ref\_idx\_lX\_active\_minus1; j++)  {  if (PicOrderCnt ( RefIdxLX[i] ) == PicOrderCnt ( BaseRefIdxLX[j]) )  {  copy WP parameters of base view to the current view;  }  }  }  }  } |

# Test sequences

Fading in/out sequences are generated as the same manner of HEVC. In JCTVC-E041[1], there are an enclosed software that it can create linear fading sequences.

In this experiment, we use same conditions as in [1], [2]: a linear fade is applied to the first 2 seconds of the 3DV sequences and the experiments are done with the 2-second sequences. A linear fade-out is applied during the first half (0~1 second) and a linear fade-in is applied during the second half(1~2 second) (Figure 3.1). The fade strength is the range [0.25, 1]. In this experiments, only texture video are encoded.

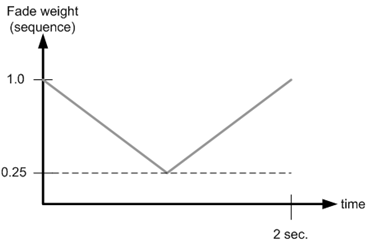


Figure 3.1 fading in/out strength

# Results

Table 2 shows the performance of the WP for the fading test sequences with respect to the HTM4.0.1. For this experiment, the WP parameters for inter-view reference picture are set to default values by encoder.

Table 2 Coding performance of the weighted prediction

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | enc time | dec time |
| Balloons | -36.0% | -19.5% | -17.5% | -29.6% | 158.0% | 100.0% |
| Kendo | -18.6% | -3.6% | -2.5% | -13.1% | 192.6% | 102.0% |
| Newspapercc | -33.0% | -22.5% | -23.9% | -29.3% | 140.7% | 93.7% |
| GhostTownFly | -29.6% | -2.5% | -5.0% | -24.8% | 162.1% | 105.0% |
| PoznanHall2 | -32.1% | -11.4% | -12.1% | -24.2% | 166.7% | 100.5% |
| PoznanStreet | -27.5% | -15.4% | -14.3% | -24.4% | 136.9% | 102.6% |
| UndoDancer | -37.2% | -19.9% | -23.8% | -33.2% | 156.2% | 102.1% |
| 1024x768 | -29.2% | -15.2% | -14.6% | -24.0% | 162.4% | 98.5% |
| 1920x1088 | -31.6% | -12.3% | -13.8% | -26.7% | 155.0% | 102.5% |
| average | -30.6% | -13.6% | -14.2% | -25.5% | 158.1% | 100.8% |

Table 3 shows the performance of the proposed algorithm for the fading test sequences with respect to the HTM4.0.1. In this case, the WP parameters for inter-view reference picture are set to default values in the WP parameter inheritance process at the decoder.

Table 3 Coding performance of the proposed algorithm

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | enc time | dec time |
| Balloons | -36.0% | -26.3% | -25.1% | -32.0% | 161.1% | 101.5% |
| Kendo | -18.6% | -12.9% | -11.9% | -16.4% | 193.4% | 102.3% |
| Newspapercc | -33.0% | -28.2% | -29.2% | -31.3% | 143.9% | 94.8% |
| GhostTownFly | -29.6% | -11.6% | -12.2% | -26.2% | 164.0% | 104.0% |
| PoznanHall2 | -32.1% | -18.6% | -19.9% | -27.0% | 168.7% | 98.6% |
| PoznanStreet | -27.5% | -19.1% | -19.4% | -25.5% | 141.8% | 101.1% |
| UndoDancer | -37.2% | -23.8% | -26.7% | -33.9% | 158.4% | 102.2% |
| 1024x768 | -29.2% | -22.5% | -22.1% | -26.6% | 164.9% | 99.5% |
| 1920x1088 | -31.6% | -18.3% | -19.6% | -28.2% | 157.9% | 101.5% |
| **average** | **-30.6%** | **-20.1%** | **-20.7%** | **-27.5%** | 160.9% | 100.6% |

Table 4 shows the performance of the proposed algorithm in table 3 with respect to the weighted prediction in Table 2.

Table 4 Coding performance of the proposed algorithm with respect to weighted prediction.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | enc time | dec time |
| Balloons | 0.0% | -8.2% | -9.1% | -3.4% | 102.0% | 101.5% |
| Kendo | 0.0% | -9.9% | -9.7% | -3.9% | 100.4% | 100.2% |
| Newspapercc | 0.0% | -7.3% | -6.8% | -2.8% | 102.3% | 101.2% |
| GhostTownFly | 0.0% | -9.2% | -8.1% | -1.9% | 101.2% | 99.0% |
| PoznanHall2 | 0.0% | -8.1% | -8.6% | -3.6% | 101.2% | 98.1% |
| PoznanStreet | 0.0% | -4.3% | -6.0% | -1.4% | 103.6% | 98.5% |
| UndoDancer | 0.0% | -4.7% | -3.6% | -1.1% | 101.4% | 100.1% |
| 1024x768 | 0.0% | -8.4% | -8.5% | -3.4% | 101.6% | 101.0% |
| 1920x1088 | 0.0% | -6.6% | -6.6% | -2.0% | 101.8% | 98.9% |
| **average** | **0.0%** | **-7.4%** | **-7.4%** | **-2.6%** | 101.7% | 99.8% |

# Conclusion

This contribution proposed that WP be enabled both for the base and the dependent views with a provision that inherits the WP parameters of a dependent view from the base view. A flag indicating whether WP parameters on the dependent view will be inherited from the independent view is introduced in the slice header. It is shown that inheritance of weighted prediction parameters further improves coding efficiency of dependent views by -7.4%, -7.4%, and -2.6% for V1, V2, and coded video, respectively. It is recommended that the proposed weighted prediction parameter inheritance method be adopted.

# Patent rights declaration(s)

**LG Electronics may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**

# References

1. Philippe Bordes, “Weighted Prediction ", JCTVC-E041, Geneva, March 2011.
2. Philippe Bordes, “Weighted Prediction ", JCTVC-F265, Torino, July 2011.

# WD text changes

Table 5 shows the changed slice header syntax for the proposed algorithm.

Table5 Syntax of slice header

|  |  |
| --- | --- |
| slice\_header( ) { | Descriptor |
| **first\_slice\_in\_pic\_flag** | u(1) |
| **…** |  |
| if( !entropy\_slice\_flag ) { |  |
| **…** | **…** |
| if( ( weighted\_pred\_flag && slice\_type = = P) | |  ( weighted\_bipred\_flag = = 1 && slice\_type = = B ) ) { |  |
| if( ViewIdx>0 && !DepthFlag ) { |  |
| **base\_pred\_weight\_table\_flag** | u(1) |
| if( base\_pred\_weight\_table\_flag == 0 ) |  |
| pred\_weight\_table( ) |  |
| } |  |
| **…** |  |
| } |  |

**base\_pred\_weight\_table\_flag** equals to 1 specifies that the weighted prediction parameter for the texture of dependent view are inherited from the texture picture of base view. When the base\_pred\_weight\_table\_flag is not present, it is inferred to be 0.

G.8.3.4.2 Initialization process for reference picture lists

The specifications in subclause 8.3.4.2 apply. If base\_pred\_weight\_table\_flag is equal to 1 the specification in subclause G.8.3.5 apply at the end of the specification in subclause 8.3.4.2.

G.8.3.5 Weighted prediction parameters inheritance

The base view picture is an inter-view reference picture of current picture whose ViewIdx is equal to 0 and let viewIdxBase be the variable ViewIdx of the base view picture.

The LumaWeightLX[ ViewIdx ], luma\_offset\_lX[ ViewIdx ], ChromaWeightLX[ ViewIdx ], ChromaOffsetLX[ ViewIdx ] are derived from the LumaWeightLX[ viewIdxBase ], luma\_offset\_lX[ viewIdxBase ], ChromaWeightLX[ viewIdxBase ], ChromaOffsetLX[ viewIdxBase ] as follows.

For X being replaced by 0 and 1, the following applies. A variable num\_ref\_idx\_lX\_active\_minus1\_base is set to the num\_ref\_idx\_lX\_minus1 of the base view picture. A reference picture list RefPicListLXBase is set to the reference picture list RefPicListLX of the base view picture.

for( rIdx=0 ; rIdx<num\_ref\_idx\_lX\_active\_minus1+1; rIdx++ )

Let refPicViewIdx be the variable ViewIdx of the RefPicListLX[ rIdx ]

for( rIdxBase = 0; rIdxBase < num\_ref\_idx\_lX\_active\_minus1\_base+1; rIdxBase ++ )

if( refPicViewIdx != ViewIdx )

LumaWeightLX[ ViewIdx ][ rIdx ] is set 2luma\_log2\_weight\_denom

ChromaWeightLX[ ViewIdx ][ rIdx ][ 0 ] is set to 2ChromaLog2WeightDenom

ChromaWeightLX[ ViewIdx ][ rIdx ][ 1 ] is set to 2ChromaLog2WeightDenom

luma\_offset\_lX[ ViewIdx ][ rIdx ], ChromaOffsetLX[ ViewIdx ][ rIdx ][ 0 ], and ChromaOffsetLX[ ViewIdx ][ rIdx ][ 1 ] are set to 0

Otherwise, if PicOrderCnt( RefPicListLX[ rIdx ] is equal to PicOrderCnt( RefPicListXBase[ rIdxBase ]

LumaWeightLX[ ViewIdx ][ rIdx ] is set to LumaWeightLX[ rIdxBase ] of the base view picture

luma\_offset\_lX[ ViewIdx ][ rIdx ] is set to luma\_offset\_lX[ rIdxBase ] of the base view picture

ChromaWeightLX [ ViewIdx ][ rIdx ][ 0 ] is set to ChromaWeightLX[ rIdxBase ][ 0 ] of the base view picture

ChromaWeightLX [ ViewIdx ][ rIdx ][ 1 ] is set to ChromaWeightLX[ rIdxBase ][ 1 ] of the base view picture

G.8.5.2.2 Decoding process for inter prediction samples

The specifications in subclause 8.5.2.2 apply with the following modification:

– All invocations of the process specified in subclause 8.5.2.2.3 are replaced with invocations of the process specified in subclause G.8.5.2.2.3

Following additional operation apply at the end of the specification in subclause 8.5.2.2.

When res\_pred\_flag is equal to 1, the inter-view residual prediction process as specified in subclause G.8.5.2.3 is invoked with the luma location ( xB, yB ), the variables nPSW and nPSH, the reference view index refViewIdx set equal to 0, and the arrays predSamplesL, predSamplesCb, and predSamplesCr as the inputs and the outputs are modified versions of the arrays predSamplesL, predSamplesCb, and predSamplesCr.

G.8.5.2.2.3  Weighted sample prediction process

The specifications in subclause 8.5.2.2.3 apply with the following modification:

– All invocations of the process specified in subclause 8.5.2.2.3.2 are replaced with invocations of the process specified in subclause G.8.5.2.2.3.2

G.8.5.2.2.3.2 Weighted sample prediction process

Inputs to this process are:

– a location ( xB, yB ) specifying the top-left sample of the current prediction block relative to the top left sample of the current coding block,

– the width and height of this prediction block, nPbW and nPbH,

– two (nPbW)x(nPbH) arrays predSamplesL0 and predSamplesL1,

– prediction list utilization flags, predFlagL0 and predFlagL1,

– reference indices, refIdxL0 and refIdxL1,

– a variable cIdx specifying colour component index,

– a bit depth of samples, bitDepth.

Outputs of this process are:

– the (nPbW)x(nPbH) array predSamples of prediction sample values.

The variables shift1 is set equal to 14 – bitDepth.

The variables log2WD, o0, o1, and w0, w1 are derived as follows.

– If cIdx is equal to 0 for luma samples,

log2WD = luma\_log2\_weight\_denom + shift1 (8‑216)

w0 = LumaWeightL0[ ViewIdx ][ refIdxL0 ] (8‑217)

w1 = LumaWeightL1[ ViewIdx ][ refIdxL1 ] (8‑218)

o0 = luma\_offset\_l0[ ViewIdx ][ refIdxL0 ] \* ( 1 << ( bitDepth − 8 ) ) (8‑219)

o1 = luma\_offset\_l1[ ViewIdx ][ refIdxL1 ] \* ( 1 << ( bitDepth − 8 ) ) (8‑220)

– Otherwise (cIdx is not equal to 0 for chroma samples),

log2WD = ChromaLog2WeightDenom + shift1 (8‑221)

w0 = ChromaWeightL0[ ViewIdx ][ refIdxL0 ][ cIdx − 1 ] (8‑222)

w1 = ChromaWeightL1[ ViewIdx ][ refIdxL1 ][ cIdx − 1 ] (8‑223)

o0 = ChromaOffsetL0[ ViewIdx ][ refIdxL0 ][ cIdx − 1 ] \* ( 1 << ( bitDepth − 8 ) ) (8‑224)

o1 = ChromaOffsetL1[ ViewIdx ][ refIdxL1 ][ cIdx − 1 ] \* ( 1 << ( bitDepth − 8 ) ) (8‑225)

The prediction sample predSamples[ x ][ y ] with x = 0..(nPbW)−1 and y = 0..(nPbH)−1 are derived as follows:

– If the predFlagL0 is equal to 1 and predFlagL1 is equal to 0, the prediction samples are derived by:

if( log2WD >= 1 )   
 predSamples[ x ][ y ] = Clip3( 0, ( 1 << bitDepth ) − 1,   
 ( (predSamplesL0[ x ][ y ] \* w0 + 2log2WD − 1) >> log2WD ) + o0 ) (8‑226)  
else  
 predSamples[ x ][ y ] = Clip3( 0, ( 1 << bitDepth ) − 1, predSamplesL0[ x ][ y ] \* w0 + o0 ) (8‑227)

– Otherwise, if the predFlagL0 is equal to 0 and predFlagL1 is equal to 1, the final predicted sample values predSamples [ x ][ y ] are derived by

if( log2WD >= 1 )  
 predSamples[ x ][ y ] = Clip3( 0, ( 1 << bitDepth ) − 1,   
 ( (predSamplesL1[ x ][ y ] \* w1 + 2log2WD − 1) >> log2WD ) + o1 ) (8‑228)  
else  
 predSamples[ x ][ y ] = Clip3( 0, ( 1 << bitDepth ) − 1, predSamplesL1[ x ][ y ] \* w1 + o1 ) (8‑229)

– Otherwise, the final predicted sample values predSamples[ x ][ y ] are derived by

predSamples[ x ][ y ] = Clip3( 0, ( 1 << bitDepth ) − 1,   
 ( predSamplesL0 [ x ][ y ] \* w0 + predSamplesL1[ x ][ y ] \* w1 +   
 ((o0 + o1 + 1) << log2WD) ) >> (log2WD + 1) ) (8‑230)