|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  9th Meeting: Geneva, CH, 27 April – 7 May 2012 | Document: JCTVC-I0353r1 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Hooks for temporal motion vector prediction and weighted prediction in HEVC multiview/3DV extension** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Ying Chen Ye-Kui Wang Li Zhang Vadim Seregin  Jianle Chen  5775 Morehouse Dr San Diego, CA 92121 USA | Tel: Email: | 1-858-845-6589 [cheny@qualcomm.com](mailto:cheny@qualcomm.com)  1-858-651-8345 [yekuiw@qualcomm.com](mailto:yekuiw@qualcomm.com)  1-858-651-6660 [lizhang@qualcomm.com](mailto:lizhang@qualcomm.com)  1-858-651-3481 [vseregin@qualcomm.com](mailto:vseregin@qualcomm.com)  1-858-651-8028 [cjianle@qualcomm.com](mailto:cjianle@qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

In multiview or 3D Video (3DV) extension of HEVC under development, it is possible that there would be a profile wherein only high-level syntax only changes are introduced compared to HEVC (base spec), similarly as the existing AVC based MVC extension compared to AVC (Annex A profiles). In HEVC, both Temporal Motion Vector Prediction (TMVP) and implicit weighted prediction are designed in a way that POC distances need to be checked. However, in multiview or 3DV, a picture from a different view, i.e., view component in the context of AVC based MVC, may be present in a reference picture list of the current picture (view component). In this case, a picture and one of its reference pictures can have the same POC value. Zero POC distance is a problem for both POC based motion vector scaling (e.g., in TMVP) and POC based implicit prediction weights calculation (in weighted prediction).

It is proposed that hooks with slight modifications in HEVC base spec can be provided, in order to avoid the above problem while keeping the benefits of TMVP and weighted prediction in the multivew or 3DV extension of HEVC being developed in MPEG.

# Introduction

## Multiview video coding

In MPEG HTM (HEVC based 3DV Test Model), one view can be predicted from the other by the so-called disparity motion compensation. It is realized in a way that a decoded view component from a different view but in the same time instance can be added in a reference picture list of the current view component. For example, as shown in Figure 1, wherein the vertical indices (V1, V0 and V2) of the pictures correspond to view identifiers and the horizontal indices (T0 through T11) correspond to POC values. Each square is a view component and when a vertical arrow links view component A (e.g., T5/V0) to another view component B (e.g., T5/V1, in the same time instance), in a way that view component A can be added into the reference picture list 0 of the view component B.

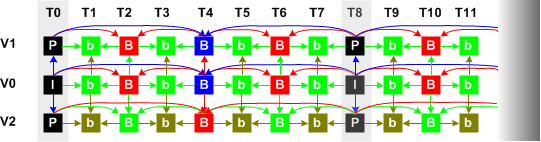


Figure 1: Typical motion prediction structure in 3DV (texture only).

Note that in the above example, both view component A and view component B have the same POC value (5).

A motion vector pointing to a view component of a different view is called disparity motion vector in this document and this view component is referred to as an inter-view reference picture.

## MVP

In multiview or 3D Video (3DV) extension of HEVC, when a motion vector A in a reference picture is to be used as an AMVP or MERGE candidate for the prediction of motion vector B (of the current PU) and only one of them is a disparity motion vector, the motion vector scaling as defined in sub-clause 8.5.2.1.6 fails, since it either return a zero motion vector (coding efficiency decreases) or even couldn’t be performed due to the division by a zero value.

When a motion vector A from a spatially neighbouring block is to be used as an AMVP candidate for the prediction of motion vector B (of the current PU) and only one of them is a disparity motion vector, similarly if either motion vector A or motion vector B is a disparity motion vector, the motion vector scaling process fails.

## Implicit weighted prediction

When a reference picture pair of a bi-predicted PU consists of a temporal reference picture and an inter-view reference picture, weighted prediction based on the weights calculated per the current HEVC design is equivalent to uni-directionally predicting from either the temporal reference picture or the inter-view reference picture.

When a reference picture pair of the bi-predicted PU consists of two inter-view reference pictures, the scaling based on POC distance (as in e.g., sub-clause 8.5.2.2.3.2) fails due to the division of zero.

# Proposal

To solve the above problems, it is proposed that the current HEVC draft is changed to enable forward compatibility, which allows for a profile in the multiview or 3DV extension of HEVC to have high-level syntax changes only compared to the HEVC base spec, and to solve the problems on motion vector prediction and implicit weighted prediction involving disparity motion vectors. There is no syntax or semantics change for the current HEVC specification and has no impact on the current HEVC bitstreams.

## MVP

It is proposed that a motion vector belonging to a different category (e.g., disparity motion vector) is not used to predict a normal motion vector (temporal motion vector, with a default category indication of 0) and a normal motion vector is not used to predict a motion vector belonging to a different category (e.g., disparity motion vector). Besides, a motion vectors belonging to different category that is different from temporal motion vector is not scaled.

In the current HEVC spec. all the motion vectors belong to a same category of temporal motion vector and have a category indication of 0.

## Implicit weighted prediction

It is proposed that when one or two reference pictures of the current PU doesn’t belong to category 0, , and implicit weighted prediction mode is on, the weights for these two reference pictures of the current PU is set to be the same: (½, ½).

## Derivation of the properties of RPS subsets

For each RPS subset of RefPicSetLtCurr, RefPicSetLtFoll, RefPicSetStCurrBefore, RefPicSetStCurrAfter and RefPicSetStFoll, a RefTypeIdc is derived to be equal to 0.

Each picture included in an RPS subset has RefPicTypeIdc set equal to RefTypeIdc of the RPS subset.

As an example use of this in the potential MVC extension of HEVC, the RefTypeIdc of the InterView RPS subset can be set equal to 1.

## Decoding processes

The function RefPicTypeFunc( pic ) is defined, which returns the RefPicTypeIdc value of the reference picture pic.

### For motion vector prediction

8.5.2.1.6 Derivation process for motion vector predictor candidates

**…**

The motion vector mvLXA and the availability flag availableFlagLXA are derived in the following ordered steps:

1. …
2. ...
3. ...
4. …
5. When availableFlagLXA is equal to 0, for ( xAk, yAk ) from ( xA0, yA0 ) to ( xA1, yA1 ) where yA1 = yA0 - MinPuSize, the following applies repeatedly until availableFlagLXA is equal to 1:

* If the prediction unit covering luma location ( xAk, yAk ) is available, PredMode is not MODE\_INTRA, predFlagLX[ xAk ][ yAk ] is equal to 1, availableFlagLXA is set equal to 1, the motion vector mvLXA is set equal to the motion vector mvLX[ xAk ][ yAk ], refIdxA is set equal to refIdxLX[ xAk ][ yAk ], ListA is set equal to LX.
* Otherwise if the prediction unit covering luma location ( xAk, yAk ) is available, PredMode is not MODE\_INTRA, predFlagLY[ xAk ][ yAk ] (with Y = !X) is equal to 1, availableFlagLXA is set equal to 1, the motion vector mvLXA is set equal to the motion vector mvLY[ xAk ][ yAk ], refIdxA is set equal to refIdxLY[ xAk ][ yAk ], ListA is set equal to LY.
* If availableFlagLXA is equal to 1, and RefPicTypeFunc ( RefPicListListA( refIdxA ) ) is not equal to RefPicTypeFunc ( RefPicListLX( refIdxLX ) ), availableFlagLXA is set to 0.
* When availableFlagLXA is equal to 1, and RefPicTypeFunc ( RefPicListListA( refIdxA ) ) and RefPicTypeFunc ( RefPicListLX( refIdxLX ) ) are both equal to 0, mvLXA is derived as specified below.

tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (8‑136)

DistScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (8‑137)

mvLXA = Clip3( −8192, 8191.75, Sign( DistScaleFactor \* mvLXA ) \*    
 ( (Abs( DistScaleFactor \* mvLXA ) + 127 ) >> 8 ) ) (8‑138)

where td and tb are derived as

td = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListListA( refIdxA ) ) ) (8‑139)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListLX( refIdxLX ) ) ) (8‑140)

* When availableFlagLXA is equal to 1, and RefPicTypeFunc( RefPicListListA( refIdxA ) ) and RefPicTypeFunc( RefPicListLX( refIdxLX ) ) are both equal to a non-zero value, mvLXA is set to mvLXA without scaling.

The motion vector mvLXB and the availability flag availableFlagLXB are derived in the following ordered steps:

1. ...
2. …
3. ...
4. …
5. …
6. When isScaledFlagLX is equal to 0, availableFlagLXB is set equal to 0 and for ( xBk, yBk ) from ( xB0, yB0 ) to ( xB2, yB2 ) where xB0 = xP +nPSW, xB1 = xB0 - MinPuSize , and xB2 =  xP - MinPuSize, the following applies repeatedly until availableFlagLXB is equal to 1:

* If the prediction unit covering luma location ( xBk, yBk ) is available, PredMode is not MODE\_INTRA, predFlagLX[ xBk ][ yBk ] is equal to 1, availableFlagLXB is set equal to 1, the motion vector mvLXB is set equal to the motion vector mvLX[ xBk ][ yBk ], refIdxB is set equal to refIdxLX[ xBk ][ yBk ], ListB is set equal to LX.
* Otherwise if the prediction unit covering luma location ( xBk, yBk ) is available, PredMode is not MODE\_INTRA, predFlagLY[ xBk ][ yBk ] (with Y = !X) is equal to 1, availableFlagLXB is set equal to 1, the motion vector mvLXB is set equal to the motion vector mvLY[ xBk ][ yBk ], refIdxB is set equal to refIdxLY[ xBk ][ yBk ], ListB is set equal to LY.
* If availableFlagLXB is equal to 1, and RefPicTypeFunc ( RefPicListListB( refIdxB ) ) is not equal to RefPicTypeFunc ( RefPicListLX( refIdxLX ) ), availableFlagLXB is set to 0.
* When availableFlagLXB is equal to 1 and RefPicTypeFunc ( RefPicListListB( refIdxB ) ) and RefPicTypeFunc ( RefPicListLX( refIdxLX ) ) are both equal to 0, and PicOrderCnt( RefPicListListB( refIdxB ) ) is not equal to PicOrderCnt( RefPicListLX( refIdxLX ) ), mvLXB is derived as specified below.

tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (8‑144)

DistScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (8‑145)

mvLXB =Clip3( −8192, 8191.75, Sign( DistScaleFactor \* mvLXA ) \*   
 ( (Abs( DistScaleFactor \* mvLXA ) + 127 ) >> 8 ) ) (8‑146)

where td and tb are derived as

td = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListListB( refIdxB ) ) ) (8‑147)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListLX( refIdxLX ) ) ) (8‑148)

* When availableFlagLXB is equal to 1, and RefPicTypeFunc ( RefPicListListB( refIdxB ) ) and RefPicTypeFunc ( RefPicListLX( refIdxLX ) ) are both equal to a non-zero value, mvLXB is set to mvLXB without scaling.

8.5.2.1.7 Derivation process for temporal luma motion vector prediction

**…**

The variables mvLXCol and availableFlagLXCol are derived as follows.

* If one of the following conditions is true, both components of mvLXCol are set equal to 0 and availableFlagLXCol is set equal to 0.
  + colPu is coded in an intra prediction mode.
  + colPu is marked as "unavailable".
  + colPic is marked as "unused for temporal motion vector prediction".
  + enable\_temporal\_mvp\_flag is equal to 0.
* Otherwise, the motion vector mvCol, the reference index refIdxCol, and the reference list identifier listCol are derived as follows.
  + If PredFlagL0[ xPCol ][ yPCol ] is equal to 0, mvCol, refIdxCol, and listCol are set equal to MvL1[ xPCol ][ yPCol ], RefIdxL1[ xPCol ][ yPCol ], and L1, respectively.
  + Otherwise (PredFlagL0[ xPCol ][ yPCol ] is equal to 1), the following applies.
  + If PredFlagL1[ xPCol ][ yPCol ] is equal to 0, mvCol, refIdxCol, and listCol are set equal to MvL0[ xPCol ][ yPCol ], RefIdxL0[ xPCol ][ yPCol ], and L0, respectively.
  + Otherwise (PredFlagL1[ xPCol ][ yPCol ] is equal to 1), the following assignments are made.
    - * If PicOrderCnt( pic ) of every picture pic in every reference picture lists is less than or equal to PicOrderCntVal, mvCol, refIdxCol, and listCol are set equal to MvLX[ xPCol ][ yPCol ], RefIdxLX[ xPCol ][ yPCol ] and LX, respectively with X being the value of X this process is invoked for.
      * Otherwise (PicOrderCnt( pic ) of at least one picture pic in at least one reference picture list is greater than PicOrderCntVal, mvCol, refIdxCol and listCol are set equal to MvLN[ xPCol ][ yPCol ], RefIdxLN[ xPCol ][ yPCol ] and LN, respectively with N being the value of collocated\_from\_l0\_flag.

and the variable availableFlagLXCol is set equal to 1 and the following applies.

* + If RefPicTypeFunc( RefPicListLX ( refIdxLX ) ) is not equal to RefPicTypeFunc( listCol( refIdxCol ) ), availableFlagLXCol is set equal to 0. Note that listCol( refIdxCol ) returns returns the reference picture with index refIdxCol from reference picture list listCol of the picture colPic.
  + If availableFlagLXCol is 1 and RefPicTypeFunc ( RefPicListLX( refIdxLX ) ) and RefPicTypeFunc ( listCol( refIdxCol ) ) are both equal to a non-zero value, or PicOrderCnt( colPic ) – RefPicOrderCnt( colPic, refIdxCol, listCol ) is equal to PicOrderCntVal - PicOrderCnt( RefPicListLX ( refIdxLX ) ),

mvLXCol = mvCol (8‑153)

* + Otherwise, if RefPicTypeFunc( RefPicListLX( refIdxLX ) ) and RefPicTypeFunc( listCol( refIdxCol ) ) are both equal to 0, mvLXCol is derived as scaled version of the motion vector mvCol as specified below

tx = ( 16384 + ( Abs( td ) >>1 ) ) / td (8‑154)

DistScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (8‑155)

mvLXCol =  Clip3( −8192, 8191.75, Sign( DistScaleFactor \* mvCol ) \*    
 ( (Abs( DistScaleFactor \* mvCol ) + 127 ) >> 8 ) ) (8‑156)

where td and tb are derived as

td = Clip3( −128, 127, PicOrderCnt( colPic ) – RefPicOrderCnt( colPic, refIdxCol, listCol ) ) (8‑157)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListLX ( refIdxLX ) ) ) (8‑158)

### For implicit weighted prediction

8.5.2.2.3 Weighted sample prediction process

Inputs to this process are:

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

– the width and height of this prediction unit, nPSW and nPSH,

– two (nPSW)x(nPSH) arrays predSamplesL0 and predSamplesL1,

– prediction list utilization flags, predFlagL0 and predFlagL1,

– reference indices, refIdxL0 and refIdxL1,

– motion vectors, mvL0 and mvL1,

– the bit-depth of the chroma component, bitDepth.

Outputs of this process are:

– the (nPSW)x(nPSH) array predSamples of prediction sample values.

Variables shift1, shift2, offset1 and offset2 are derived as follows.

– The variable shift1 is set equal to 14 – bitDepth and the variable shift2 is set equal to 15 – bitDepth,

– The variable offset1 is set equal to 1 << ( shift1 − 1 ) and the variable offset2 is set equal to 1 << ( shift2 − 1 ).

In P slices, if the value of predFlagL0 is equal to 1, the following applies.

– If weighted\_pred\_flag is equal to 0, the default weighted sample prediction process as described in subclause 8.5.2.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause.

– Otherwise (weighted\_pred\_flag is equal to 1), the explicit weighted sample prediction process as described in subclause 8.5.2.2.3.2 is invoked with the same inputs and outputs as the process described in this subclause.

In B slices, if predFlagL0 or predFlagL1 is equal to 1, the following applies.

– If weighted\_bipred\_idc is equal to 0, the default weighted sample prediction process as described in subclause 8.5.2.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause.

– Otherwise, if weighted\_bipred\_idc is equal to 1 and if predFlagL0 or predFlagL1 equal to 1, the explicit weighted sample prediction process as described in subclause 8.5.2.2.3.2 is invoked with the same inputs and outputs as the process described in this subclause.

– Otherwise (weighted\_bipred\_idc is equal to 2), the following applies.

– If predFlagL0 is equal to 1 and predFlagL1 is equal to 1, and both RefPicTypeFunc( RefPicListL0( refIdxL0 ) ) and RefPicTypeFunc( RefPicListL1( refIdxL1 ) ) are equal to 0, the implicit weighted sample prediction process as described in subclause 8.5.2.2.3.2 is invoked with the same inputs and outputs as the process described in this subclause.

– Otherwise (predFlagL0 or predFlagL1 are equal to 1 but not both), the default weighted sample prediction process as described in subclause 8.5.2.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause.

8.5.2.2.3.1. Default weighted sample prediction process

Inputs to this process are:

– the same as specified in subclause 8.5.2.2.3.

Outputs of this process are:

– the same as specified in subclause 8.5.2.2.3.

Depending on the value of predFlagL0 and predFlagL1, the prediction samples predSamples[ x, y ] with x = 0..(nPSW)−1 and y = 0..(nPSH)−1 are derived as follows.

– If predFlagL0 is equal to 1 and predFlagL1 is equal to 0,

predSamples[ x, y ] = Clip3( 0, ( 1 << bitDepth ) − 1, ( predSamplesL0[ x, y ] + offset1 ) >> shift1 ) (8‑211)

– Otherwise, if predFlagL0 is equal to 0 and predFlagL1 is equal to 1,

predSamples[ x, y ] = Clip3( 0, ( 1 << bitDepth ) − 1, ( predSamplesL1[ x, y ] + offset1 ) >> shift1 ) (8‑212)

– Otherwise (both predFlagL0 and predFlagL1 are equal to 1), if RefPicOrderCnt( currPic, refIdxL0, L0) is equal to RefPicOrderCnt( currPic, refIdxL1, L1) and mvL0 is equal to mvL1 and both RefPicTypeFunc( RefPicListL0( refIdxL0 ) ) and RefPicTypeFunc( RefPicListL1( refIdxL1 ) ) are equal to 0,

predSamples[ x, y ] = Clip3( 0, ( 1 << bitDepth ) − 1, ( predSamplesL0[ x, y ] + offset1 ) >> shift1 ) (8‑213)

– Otherwise,

predSamples[ x, y ] = Clip3( 0, ( 1 << bitDepth ) − 1 ,   
 ( predSamplesL0[ x, y ] + predSamplesL1[ x, y ] + offset2 ) >> shift2 ) (8‑214)

# Patent rights declaration(s)

**Qualcomm Incorporated 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).**