#### 8.5.6.3 Fractional sample interpolation process

#### 8.5.6.3.1 General

Inputs to this process are:

* a luma location ( xSb, ySb ) specifying the top-left sample of the current coding subblock relative to the top‑left luma sample of the current picture,
* a variable sbWidth specifying the width of the current coding subblock,
* a variable sbHeight specifying the height of the current coding subblock,
* a motion vector offset mvOffset,
* a refined motion vector refMvLX,
* the selected reference picture sample array refPicLX,
* the bidirectional optical flow flag bdofFlag,
* a variable cIdx specifying the colour component index of the current block.

Outputs of this process are:

* an (sbWidth +bdofOffset)x(sbHeight +bdofOffset) array predSamplesLX of prediction sample values.

The bidirectional optical flow boundary offset bdofOffset is derived as follows:

bdofOffset = bdofFlag ? 2 : 0 (8‑720)

* If cIdx is equal to 0, the following applies:
  + Let ( xIntL, yIntL ) be a luma location given in full-sample units and ( xFracL, yFracL ) be an offset given in 1/16-sample units. These variables are used only in this clause for specifying fractional-sample locations inside the reference sample arrays refPicLX.
  + For each luma sample location ( xL = 0..sbWidth − 1 +bdofOffset, yL = 0..sbHeight − 1 +bdofOffset ) inside the prediction luma sample array predSamplesLX, the corresponding prediction luma sample value predSamplesLX[ xL ][ yL ] is derived as follows:
* The variables xIntL, yIntL, xFracL and yFracL are derived as follows:

xIntL = xSb + ( refMvLX[ 0 ]  >>  4 ) + xL (8‑721)

yIntL = ySb + ( refMvLX[ 1 ]  >>  4 ) + yL (8‑722)

xFracL = refMvLX[ 0 ] & 15 (8‑723)

yFracL = refMvLX[ 1 ] & 15 (8‑724)

* + If bdofFlag is equal to TRUE and one or more of the following conditions are true, the prediction luma sample value predSamplesLX[ xL ][ yL ] is derived by invoking the luma integer sample fetching process as specified in clause 8.5.6.3.3 with ( xIntL, yIntL ), ( xFracL, yFracL ) and refPicLX as inputs:
    - xL is equal to 0.
    - xL is equal to sbWidth + 1.
    - yL is equal to 0.
    - yL is equal to sbHeight + 1.
  + Otherwise, the following applies:
    - The motion vector mvLX is set equal to ( refMvLX − mvOffset ).
    - The list padVal[ dir ] is derived as follows for dir = 0..1:
      * The variable disp is derived as follows:

disp = ( refMvLX[ dir ]  >>  4) − ( mvLX[ dir ]  >>  4) + ( dir = = 0 ? xL : yL) (8‑725)

* + - * If disp is less than 0, padVal[ dir ] is set equal to disp.
      * Otherwise, if disp is greater than ( dir  = = 0  ?  sbWidth : sbHeight ) − 1, padVal[ dir ] is set equal to disp − ( ( dir  = =  0  ?  sbWidth : sbHeight ) − 1).
      * Otherwise, padVal[ dir ] is set equal to 0.
    - The prediction luma sample value predSamplesLX[ xL ][ yL ] is derived by invoking the luma sample 8-tap interpolation filtering process as specified in clause 8.5.6.3.2 with ( xIntL − bdofFlag, yIntL − bdofFlag), ( xFracL, yFracL ), refPicLX, sbWidth, sbHeight, ( xSb, ySb ) and padVal as inputs.
* Otherwise (cIdx is not equal to 0), the following applies:
  + Let ( xIntC, yIntC ) be a chroma location given in full-sample units and ( xFracC, yFracC ) be an offset given in 1/32 sample units. These variables are used only in this clause for specifying general fractional-sample locations inside the reference sample arrays refPicLX.
  + For each chroma sample location ( xC = 0..sbWidth − 1, yC = 0.. sbHeight − 1 ) inside the prediction chroma sample arrays predSamplesLX, the corresponding prediction chroma sample value predSamplesLX[ xC ][ yC ] is derived as follows:
* The variables xIntC, yIntC, xFracC and yFracC are derived as follows: [Ed. (SL): Shall we make it consistent: /2 or /SubWidthC and /SubHeightC?]

xIntC = ( xSb / SubWidthC ) + ( mvLX[ 0 ]  >>  5 ) + xC (8‑726)

yIntC = ( ySb / SubHeightC ) + ( mvLX[ 1 ]  >>  5 ) + yC (8‑727)

xFracC = mvLX[ 0 ] & 31 (8‑728)

yFracC = mvLX[ 1 ] & 31 (8‑729)

* + The motion vector mvLX is set equal to ( refMvLX − mvOffset ).
  + The list padVal[ dir ] is derived as follows for dir = 0..1:
    - The variable disp is derived as follows:

disp = ( refMvLX[ dir ]  >>  4) − ( mvLX[ dir ]  >>  4) + ( dir = = 0 ? xC : yC) (8‑730)

* + - If disp is less than 0, padVal[ dir ] is set equal to disp.
    - Otherwise, if disp is greater than ( dir  = = 0  ?  sbWidth / SubWidthC : sbHeight / SubWidthC ) − 1, padVal[ dir ] is set equal to disp − ( ( dir  = =  0  ?  sbWidth / SubWidthC : sbHeight / SubWidthC ) − 1).
    - Otherwise, padVal[ dir ] is set equal to 0.
  + The prediction sample value predSamplesLX[ xC ][ yC ] is derived by invoking the process specified in clause 8.5.6.3.4 with ( xIntC, yIntC ), ( xFracC, yFracC ), refPicLX, and padVal as inputs.

##### 8.5.6.3.2 Luma sample interpolation filtering process

Inputs to this process are:

– a luma location in full-sample units ( xIntL, yIntL ),

– a luma location in fractional-sample units ( xFracL, yFracL ),

– the luma reference sample array refPicLXL,

* a variable sbWidth specifying the width of the current subblock,
* a variable sbHeight specifying the height of the current subblock,
* a luma location ( xSb, ySb ) specifying the top-left sample of the current subblock relative to the top‑left luma sample of the current picture,

– a list padVal[ dir ] with dir = 0,1 specifying reference sample padding direction and amount.

Output of this process is a predicted luma sample value predSampleLXL

The variables shift1, shift2 and shift3 are derived as follows:

– The variable shift1 is set equal to Min( 4, BitDepthY − 8 ), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max( 2, 14 − BitDepthY ).

– The variable picW is set equal to pic\_width\_in\_luma\_samples and the variable picH is set equal to pic\_height\_in\_luma\_samples.

The luma interpolation filter coefficients fL[ p ] for each 1/16 fractional sample position p equal to xFracL or  yFracL are specified in Table 8‑13.

The luma interpolation filter coefficients fPadL[ 0 ] are set equal to fL[ xFracL ] and fPadL[ 1 ] are set equal to fL[ yFracL ] and modified depending on padVal[ dir ] as follows for dir being equal to 0 and 1:

– If padVal[ dir ] is equal to − 2, fPadL[ dir ] is modified as follows:

fPadL[ dir ][ 2 ] = fPadL[ dir ][ 2 ] + fPadL[ dir ][ 1 ] + fPadL[ dir ][ 0 ] (8‑731)

fPadL[ dir ][ 0 ] = 0 (8‑732)

fPadL[ dir ][ 1 ] = 0 (8‑733)

– Otherwise, if padVal[ dir ] is equal to − 1, fPadL[ dir ] is modified as follows:

fPadL[ dir ][ 1 ] = fPadL[ dir ][ 1 ] + fPadL[ dir ][ 0 ] (8‑734)

fPadL[ dir ][ 0 ] = 0 (8‑735)

– Otherwise, if padVal[ dir ] is equal to 1, fPadL[ dir ] is modified as follows:

fPadL[ dir ][ 6 ] = fPadL[ dir ][ 6 ] + fPadL[ dir ][ 7 ] (8‑736)

fPadL[ dir ][ 7 ] = 0 (8‑737)

– Otherwise, if padVal[ dir ] is equal to 2, fPadL[ dir ] is modified as follows:

fPadL[ dir ][ 5 ] = fPadL[ dir ][ 5 ] + fPadL[ dir ][ 6 ] + fPadL[ dir ][ 7 ] (8‑738)

fPadL[ dir ][ 6 ] = 0 (8‑739)

fPadL[ dir ][ 7 ] = 0 (8‑740)

When MotionModelIdc[ xSb ][ ySb ] is greater than 0 [Ed. (JC): the condidtion of affine block seems unnecessary since 4x4 normal inter block does not exist anymore] and sbWidth is equal to 4 and sbHeight is equal to 4, fPadL[ dir ] is modified as follows for dir being equal to 0 and 1:

fPadL[ dir ][ 1 ] = fPadL[ dir ][ 0 ] + fPadL[ dir ][ 1 ] (8‑741)

fPadL[ dir ][ 0 ] = 0 (8‑742)

fPadL[ dir ][ 6 ] = fPadL[ dir ][ 6 ] + fPadL[ dir ][ 7 ] (8‑743)

fPadL[ dir ][ 7 ] = 0 (8‑744)

The luma locations in full-sample units ( xInti, yInti ) are derived as follows for i = 0..7:

xInti = Clip3( 0, picW − 1, sps\_ref\_wraparound\_enabled\_flag ?  
 ClipH( ( sps\_ref\_wraparound\_offset\_minus1 + 1 ) \* MinCbSizeY, picW, xIntL + i − 3 ) : (8‑745)   
 xIntL + i − 3 )

yInti = Clip3( 0, picH − 1, yIntL + i − 3 ) (8‑746)

The predicted luma sample value predSampleLXL is derived as follows:

– If both xFracLand yFracL are equal to 0, the value of predSampleLXL is derived as follows:

predSampleLXL = refPicLXL[ xInt3 ][ yInt3 ] << shift3 (8‑747)

– Otherwise, if xFracL is not equal to 0 and yFracL is equal to 0, the value of predSampleLXL is derived as follows:

predSampleLXL =   >>  shift1 (8‑748)

– Otherwise, if xFracL is equal to 0 and yFracL is not equal to 0, the value of predSampleLXL is derived as follows:

predSampleLXL =   >>  shift1 (8‑749)

– Otherwise, if xFracL is not equal to 0 and yFracL is not equal to 0, the value of predSampleLXL is derived as follows:

* The sample array temp[ n ] with n = 0..7, is derived as follows:

temp[ n ] =   >>  shift1 (8‑750)

* The predicted luma sample value predSampleLXL is derived as follows:

predSampleLXL =   >>  shift2 (8‑751)

Table 8‑13 – Specification of the luma interpolation filter coefficients fL[ p ] for each 1/16 fractional sample position p.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fractional sample position p** | **interpolation filter coefficients** | | | | | | | |
| **fL[ p ][ 0 ]** | **fL[ p ][ 1 ]** | **fL[ p ][ 2 ]** | **fL[ p ][ 3 ]** | **fL[ p ][ 4 ]** | **fL[ p ][ 5 ]** | **fL[ p ][ 6 ]** | **fL[ p ][ 7 ]** |
| 1 | 0 | 1 | −3 | 63 | 4 | −2 | 1 | 0 |
| 2 | −1 | 2 | −5 | 62 | 8 | −3 | 1 | 0 |
| 3 | −1 | 3 | −8 | 60 | 13 | −4 | 1 | 0 |
| 4 | −1 | 4 | −10 | 58 | 17 | −5 | 1 | 0 |
| 5 | −1 | 4 | −11 | 52 | 26 | −8 | 3 | −1 |
| 6 | −1 | 3 | −9 | 47 | 31 | −10 | 4 | −1 |
| 7 | −1 | 4 | −11 | 45 | 34 | −10 | 4 | −1 |
| 8 | −1 | 4 | −11 | 40 | 40 | −11 | 4 | −1 |
| 9 | −1 | 4 | −10 | 34 | 45 | −11 | 4 | −1 |
| 10 | −1 | 4 | −10 | 31 | 47 | −9 | 3 | −1 |
| 11 | −1 | 3 | −8 | 26 | 52 | −11 | 4 | −1 |
| 12 | 0 | 1 | −5 | 17 | 58 | −10 | 4 | −1 |
| 13 | 0 | 1 | −4 | 13 | 60 | −8 | 3 | −1 |
| 14 | 0 | 1 | −3 | 8 | 62 | −5 | 2 | −1 |
| 15 | 0 | 1 | −2 | 4 | 63 | −3 | 1 | 0 |

##### 8.5.6.3.3 Luma integer sample fetching process

Inputs to this process are:

* a luma location in full-sample units ( xIntL, yIntL ),
* the luma reference sample array refPicLXL,

Output of this process is a predicted luma sample value predSampleLXL

The variable shift is set equal to Max( 2, 14 − BitDepthY ).

The variable picW is set equal to pic\_width\_in\_luma\_samples and the variable picH is set equal to pic\_height\_in\_luma\_samples.

The luma locations in full-sample units ( xInt, yInt ) are derived as follows:

xInt = Clip3( 0, picW − 1, sps\_ref\_wraparound\_enabled\_flag ? (8‑752)  
 ClipH( ( sps\_ref\_wraparound\_offset\_minus1 + 1 ) \* MinCbSizeY, picW, xIntL − 1) : xIntL − 1)

yInt = Clip3( 0, picH − 1, yIntL − 1) (8‑753)

The predicted luma sample value predSampleLXL is derived as follows:

predSampleLXL = refPicLXL[ xInt ][ yInt ] << shift3 (8‑754)

##### 8.5.6.3.4 Chroma sample interpolation process

Inputs to this process are:

– a chroma location in full-sample units ( xIntC, yIntC ),

– a chroma location in 1/32 fractional-sample units ( xFracC, yFracC ),

– the chroma reference sample array refPicLXC.

Output of this process is a predicted chroma sample value predSampleLXC

The variables shift1, shift2 and shift3 are derived as follows:

– The variable shift1 is set equal to Min( 4, BitDepthC − 8 ), the variable shift2 is set equal to 6 and the variable shift3 is set equal to Max( 2, 14 − BitDepthC ).

– The variable picWC is set equal to pic\_width\_in\_luma\_samples / SubWidthC and the variable picHC is set equal to pic\_height\_in\_luma\_samples / SubHeightC.

The chroma interpolation filter coefficients fC[ p ] for each 1/32 fractional sample position p equal to xFracC or  yFracC are specified in Table 8‑14.

The chroma interpolation filter coefficients fPadC[ 0 ] are set equal to fC[ xFracL ] and fPadC[ 1 ] are set equal to fC[ yFracL ] and modified depending on padVal[ dir ] as follows for dir being equal to 0 and 1:

– If padVal[ dir ] is equal to − 1, fPadC[ dir ] is modified as follows:

fPadC[ dir ][ 1 ] = fPadC[ dir ][ 1 ] + fPadC[ dir ][ 0 ] (8‑755)

fPadC[ dir ][ 0 ] = 0 (8‑756)

– Otherwise, if padVal[ dir ] is equal to 1, fPadC[ dir ] is modified as follows:

fPadC[ dir ][ 2 ] = fPadC[ dir ][ 2 ] + fPadC[ dir ][ 3 ] (8‑757)

fPadC[ dir ][ 3 ] = 0 (8‑758)

The variable xOffset is set equal to ( sps\_ref\_wraparound\_offset\_minus1 + 1 ) \* MinCbSizeY ) / SubWidthC.

The chroma locations in full-sample units ( xInti, yInti ) are derived as follows for i = 0..3:

xInti = Clip3( 0, picW − 1, sps\_ref\_wraparound\_enabled\_flag ? ClipH( xOffset, picWC, xIntC + i − 1 ) : (8‑759)  
 xIntC + i − 1 )

yInti = Clip3( 0, picHC − 1, yIntC + i − 1 ) (8‑760)

The predicted chroma sample value predSampleLXC is derived as follows:

– If both xFracC and yFracC are equal to 0, the value of predSampleLXC is derived as follows:

predSampleLXC = refPicLXC[ xInt1 ][ yInt1 ] << shift3 (8‑761)

– Otherwise if xFracC is not equal to 0 and yFracC is equal to 0, the value of predSampleLXC is derived as follows:

predSampleLXC =   >>  shift1 (8‑762)

– Otherwise if xFracC is equal to 0 and yFracC is not equal to 0, the value of predSampleLXC is derived as follows:

predSampleLXC =   >>  shift1 (8‑763)

– Otherwise if xFracC is not equal to 0 and yFracC is not equal to 0, the value of predSampleLXC is derived as follows:

* The sample array temp[ n ] with n = 0..3, is derived as follows:

temp[ n ] =   >>  shift1 (8‑764)

* The predicted chroma sample value predSampleLXC is derived as follows:

predSampleLXC =( fPadC[ 1][ 0 ] \* temp[ 0 ] +  
  fPadC[ 1 ][ 1 ] \* temp[ 1 ] +  
  fPadC[ 1 ][ 2 ] \* temp[ 2 ] + (8‑765)  
  fPadC[ 1 ][ 3 ] \* temp[ 3 ] ) >> shift2