WD description of JCTVC-F419 Proposal 1

The derivation process chooses 4 spatial candidates.

8.4.2.1.1 Derivation process for luma motion vectors for merge mode

This process is only invoked when PredMode is equal to MODE_ INTER and merge_flag [xP][yP] is equal to 1, where (xP, yP) specify the top-left luma sample of the current prediction unit relative to the top-left luma sample of the current picture.

Inputs of this process are

– a luma location (xP, yP) of the top-left luma sample of the current prediction unit relative to the top-left luma sample of the current picture,

– variables specifying the width and the height of the prediction unit for luma, nPSW and nPSH,

– a variable PartIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are

– the luma motion vectors mvL0 and mvL1,
– the reference indices refIdxL0 and refIdxL1,
– the prediction list utilization flags predFlagL0 and predFlagL1.

The motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, and the prediction utilization flags predFlagL0 and predFlagL1 are derived as specified by the following ordered steps:

1. The derivation process for merging candidates from neighboring prediction unit partitions in subclause 8.4.2.1.4 is invoked with luma location (xP, yP), the width and the height of the prediction unit nPSW and nPSH and the partition index PartIdx as inputs and the output is assigned to the availability flags availableFlagN, the motion vectors mvL0N and mvL1N, the reference indices refIdxL0N and refIdxL1N and the prediction list utilization flags predFlagL0N and predFlagL1N with N being replaced by A, B, C, D or E. If the number of availableFlagM (with M being replaced by A, B, C or D) which is equal to 1 in order of A, B, C, D and E is equal to 4, the availableFlagN is set equal to 0. (Note: If availableFlagA, availableFlagB, availableFlagC and availableFlagD are equal to 1, availableFlagE is set to 0.)

Amazon.com, Inc., Amazon.com Services LLC - Ex. 1008, Page 000001
IPR2024-00605 (Amazon.com, Inc., Amazon.com Services LLC v. Nokia Technologies Oy)

2. The derivation process of reference indices for temporal merging candidate in subclause 8.4.2.1.3 is invoked with luma location (xP, yP), nPSW, nPSH as the inputs and the output is directly assigned to refIdxLX.

3. The derivation process for temporal luma motion vector prediction in subclause 8.4.2.1.7 is invoked with luma location (xP, yP), refIdxLX as the inputs and with the output being the availability flag availableFlagLXCol and the temporal motion vector mvLXCol. The variables availableFlagCol and predFlagLXCol (with X being 0 or 1, respectively) are derived as specified below.

   availableFlagCol = availableFlagL0Col || availableFlagL1Col (8-1)

   predFlagLXCol = availableFlagLXCol (8-2)

4. The merging candidate list, mergeCandList, is constructed of which elements are given as specified order:

   1. A, if availableFlagA is equal to 1
   2. B, if availableFlagB is equal to 1
   3. C, if availableFlagC is equal to 1
   4. D, if availableFlagD is equal to 1
   5. E, if availableFlagE is equal to 1
   6. Col, if availableFlagCol is equal to 1

5. If several merging candidates have the motion vectors and the same reference indices, the merging candidates are removed from the list except the merging candidate which has the smallest order in the mergeCandList.

6. If the number of elements NumMergeCand within the mergeCandList is equal to 1, mergeIdx is set equal to 0, otherwise, mergeIdx is set equal to merge_idx[xP][yP].

7. The following assignments are made with N being the candidate at position mergeIdx in the merging candidate list mergeCandList (N = mergeCandList[mergeIdx]) and X being replaced by 0 or 1:

   mvLX[0] = mvLXN[0] (8-3)

   mvLX[1] = mvLXN[1] (8-4)

   refIdxLX = refIdxLXN (8-5)

   predFlagLX = predFlagLXN (8-6)

8. If all availability flags availableFlagN (with N being replaced by A, B, C, D, E or Col) are equal to 0, mergeIdx is set equal to 0 and the variables mvLX, refIdxLX and predFlagLX (with X being replaced by 0 or 1) are inferred as follows.

   If slice_type is equal to P, the following applies.

   mvLX[0] = 0 (8-7)

   mvLX[1] = 0 (8-8)

   refIdxL0 = 0 (8-9)

   refIdxL1 = -1 (8-10)

   predFlagL0 = 1 (8-11)

   prefFlagL1 = 0 (8-12)

   Otherwise (slice_type is equal to B), the following applies.

   mvLX[0] = 0 (8-13)

   mvLX[1] = 0 (8-14)

   refIdxL0 = 0 (8-15)

   refIdxL1 = 0 (8-16)

   predFlagL0 = 1 (8-17)

   prefFlagL1 = 1 (8-18)

8.4.2.1.2 Derivation process for spatial merging candidates

Inputs to this process are

– a luma location (xP, yP) specifying the top-left luma sample of the current prediction unit relative to the top-left sample of the current picture,

– variables specifying the width and the height of the prediction unit for luma, nPSW and nPSH,

– a variable PartIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are (with N being replaced by A, B, C, D or E and with X being replaced by 0 or 1)

– the availability flags availableFlagN of the neighbouring prediction units,

– the reference indices refIdxLXN of the neighbouring prediction units,

– the prediction list utilization flags predFlagLXN of the neighbouring prediction units,

– the motion vectors mvLXN of the neighbouring prediction units.

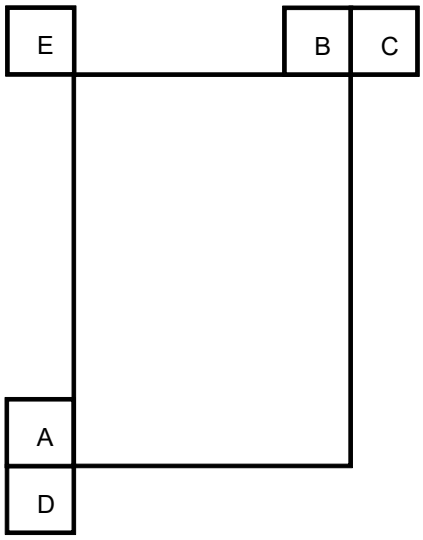


Figure 8-1 – Spatial neighbours that can be used as merging candidates (informative) illustrates the position of the spatial neighbours A, B, C, D and E relative to the current prediction unit.

For the derivation of availableFlagN, with N being A, B, C, D or E and (xN, yN) being (xP – 1, yP+nPSH-1), (xP+nPSW-1, yP – 1), (xP + nPSW, yP – 1), (xP – 1, yP+nPSH) or (xP – 1, yP – 1) the following applies.

– If one of the following conditions is true, the availableFlagN is set equal to 0, both components mvLXN are set equal to 0, refIdxLXN and predFlagLX[xN, yN] of the prediction unit covering luma location (xN, yN) are assigned respectively to mvLXN, refIdxLXN and predFlagLXN.

  – The prediction unit covering luma location (xN, yN) is not available or PredMode is MODE_INTRA.

  – PartMode of the current prediction unit is PART_2NxN and PartIdx is equal to 1 and the prediction units covering luma location (xP, yP – 1) (PartIdx = 0) and luma location (xN, yN) (Cand. N) have identical motion parameters:

    – mvLX[xP, yP – 1] == mvLX[xN, yN]

    – refIdxLX[xP, yP – 1] == refIdxLX[xN, yN]

    – predFlagLX[xP, yP – 1] == predFlagLX[xN, yN]

– PartMode of the current prediction unit is PART_Nx2N and PartIdx is equal to 1 and the prediction units covering luma location (xP – 1, yP) (PartIdx = 0) and luma location (xN, yN) (Cand. N) have identical motion parameters:

  – mvLX[xP – 1, yP] == mvLX[xN, yN]

  – refIdxLX[xP – 1, yP] == refIdxLX[xN, yN]

  – predFlagLX[xP – 1, yP] == predFlagLX[xN, yN]

– PartMode of the current prediction unit is PART_NxN and PartIdx is equal to 3 and the prediction units covering luma location (xP – 1, yP) (PartIdx = 2) and luma location (xP – 1, yP – 1) (PartIdx = 0) have identical motion parameters:

  – mvLX[xP – 1, yP] == mvLX[xP – 1, yP – 1]

  – refIdxLX[xP – 1, yP] == refIdxLX[xP – 1, yP – 1]

  – predFlagLX[xP – 1, yP] == predFlagLX[xP – 1, yP – 1]

  and the prediction units covering luma location (xP, yP – 1) (PartIdx = 1) and luma location (xN, yN) (Cand. N) have identical motion parameters:

  – mvLX[xP, yP – 1] == mvLX[xN, yN]

  – refIdxLX[xP, yP – 1] == refIdxLX[xN, yN]

  – predFlagLX[xP, yP – 1] == predFlagLX[xN, yN]

– PartMode of the current prediction unit is PART_NxN and PartIdx is equal to 3 and the prediction units covering luma location (xP, yP – 1) (PartIdx = 1) and luma location (xP – 1, yP – 1) (PartIdx = 0) have identical motion parameters:

  – mvLX[xP, yP – 1] == mvLX[xP – 1, yP – 1]

  – refIdxLX[xP, yP – 1] == refIdxLX[xP – 1, yP – 1]

  – predFlagLX[xP, yP – 1] == predFlagLX[xP – 1, yP – 1]

  and the prediction units covering luma location (xP – 1, yP) (PartIdx = 2) and luma location (xN, yN) (Cand. N) have identical motion parameters:

  – mvLX[xP – 1, yP] == mvLX[xN, yN]

  – refIdxLX[xP – 1, yP] == refIdxLX[xN, yN]

  – predFlagLX[xP – 1, yP] == predFlagLX[xN, yN]

– Otherwise, availableFlagN is set equal to 1 and the variables mvLX[xN, yN], refIdxLX[xN, yN] and predFlagLX[xN, yN] of the prediction unit covering luma location (xN, yN) are assigned respectively to mvLXN, refIdxLXN and predFlagLXN.

8.4.2.1.3 Derivation process of reference indices for temporal merging candidate

Inputs to this process are

– a luma location (xP, yP) specifying the top-left luma sample of the current prediction unit relative to the top-left sample of the current picture,

– variables specifying the width and the height of the prediction unit for luma, nPSW and nPSH.

Outputs of this process are (with X being replaced by 0 or 1)

– the reference indices refIdxLX of the current prediction units.

For the derivation of refIdxLXA, the following applies.

– If the prediction unit covering luma location (xP-1, yP+nPSH-1) is available and PredMode is not MODE_INTRA, refIdxLXA is assigned to refIdxLX[xP-1, yP+nPSH-1].

– Otherwise, refIdxLXA is assigned to -1.

For the derivation of refIdxLXB, the following applies.

– If the prediction unit covering luma location (xP+nPSW-1, yP-1) is available and PredMode is not MODE_INTRA, refIdxLXB is assigned to refIdxLX[xP+nPSW-1, yP-1].

– Otherwise, refIdxLXB is assigned to -1.

For the derivation of refIdxLXC, the following applies.

– If the prediction unit covering luma location (xP+nPSW, yP-1) is available and PredMode is not MODE_INTRA, refIdxLXC is assigned to refIdxLX[xP+nPSW, yP-1].

– Otherwise if the prediction unit covering luma location (xP-1, yP+nPSH) is available and PredMode is not MODE_INTRA, refIdxLXC is assigned to refIdxLX[xP-1, yP+nPSH].

– Otherwise if the prediction unit covering luma location (xP-1, yP-1) is available and PredMode is not MODE_INTRA refIdxLXC is assigned to refIdxLX[xP-1, yP-1].

– Otherwise, refIdxLXC is assigned to -1.

The variable refIdxLX is derived as follows.

– If refIdxLXA is equal to refIdxLXB and refIdxLXB is equal to refIdxLXC, the following applies.
  – If refIdxLXA is equal to -1 then refIdxLX = 0
  – Otherwise refIdxLX = refIdxLXA
– Otherwise if refIdxLXA is equal to refIdxLXB, the following applies.
  – If refIdxLXA is equal to -1 then refIdxLX = refIdxLXC
  – Otherwise refIdxLX = refIdxLXA
– Otherwise if refIdxLXB is equal to refIdxLXC, the following applies.
  – If refIdxLXB is equal to -1 then refIdxLX = refIdxLXA
  – Otherwise refIdxLX = refIdxLXB
– Otherwise if refIdxLXA is equal to refIdxLXC, the following applies.
  – If refIdxLXA is equal to -1 then refIdxLX = refIdxLXB
  – Otherwise refIdxLX = refIdxLXA
– Otherwise if refIdxLXA is equal to -1, the following applies.
  – refIdxLX = min(refIdxLXB, refIdxLXC)
– Otherwise if refIdxLXB is equal to -1, the following applies.
  – refIdxLX = Min(refIdxLXA, refIdxLXC)
– Otherwise if refIdxLXC is equal to -1, the following applies.
  – refIdxLX = Min(refIdxLXA, refIdxLXB)
– Otherwise, the following applies.
  – refIdxLX = Min(Min(refIdxLXA, refIdxLXB), refIdxLXC)