

Title: Coding with a single, unified reference picture list
Status: Input Document to JCT-VC
Purpose: Proposal
Author(s): David Flynn
 Nikola Šprljan
 Marta Mrak
Source: BBC Research & Development

davidf@rd.bbc.co.uk
nikola@sprljan.com
marta.mrak@bbc.co.uk

Abstract

The HM software has been modified to support coding with a single unified reference picture list. The concept of reference pairs is introduced, where a pair consists of either two references (bi-directional coding) or one reference and a null element (uni-directional coding). The pairs are ordered and a single codebook is used for coding, which avoids separate L0 and L1 indexing. The main benefit of this solution is that the selection of reference frames in bi-prediction is not restricted to all combinations of reference frames from two lists (as per WD3). Instead, we can now use a smaller, equal or larger set of reference frame combinations. In simplifying the design, this solution reduces the number of lines in both the WD and HM, while providing a marginal RD performance increase of 0.1% when the software is configured to emulate the equivalent of the HM-3.0 reference pairs.

1 Introduction

The current HM design consists of two reference lists (L0 and L1) and a combined list (LC). This design influences a number of algorithms related to inter coding. The purpose of this contribution is to solve some limitations imposed by the current design:

- selection of reference picture pairs is limited due to dependency to reference lists; adding or removing some combination of reference pictures for the codec is currently done by modifying reference lists, which in general may add or remove more pairs than needed.
- mapping direction modes for motion vectors to actual pictures from which prediction is done unnecessary depends on indexing into two reference lists, which further complicates mapping related to motion vector prediction;
- due to existence of three lists, the mapping of reference index selection to the codeword is convoluted for the CAVLC coding.

We are proposing a very simple solution that completely removes multiple reference picture lists, preserving only one list of reference frames.

2 A single list for bi-directional prediction

Figure 1 illustrates the relationship between a bi-directional PU and the reference lists L0 and L1, any single PU must pick a combination from the product of the two lists. If any combination is not used, it still occupies a code point in the entropy coding. Each list is normally ordered such that short term references are in order of increasing temporal distance from the current picture, with long term references added to the tail of the list.

By considering the combinations of the two lists as pairs of (ref_a, ref_b) , where MV_0 and MV_1 use ref_a and ref_b respectively, it is possible to describe a set of reference pairs available to the slice. An alternative is to dispense with the two lists, and considering a subset of all combinations of pictures from the reference buffer, as shown in figure 2, where used references are of POC 0, 4 and 8. This subset can either be signalled to the decoder by naming each pair, or by sending the parameters needed for their construction according to a predefined method.

Figure 1: The relationship between a bi-directional PU and the reference pictures

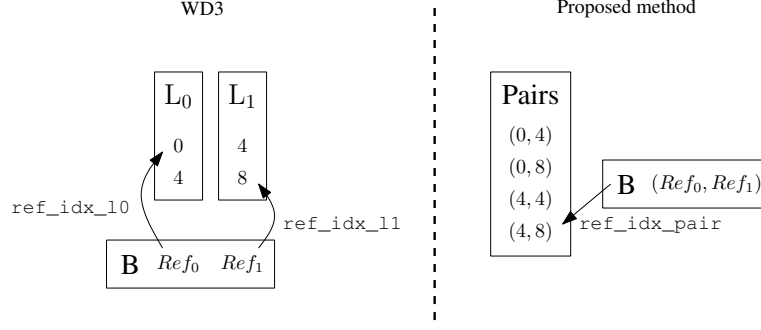
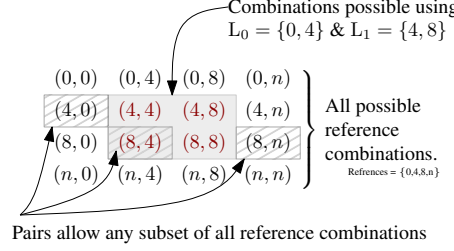


Figure 2: Illustration of possible reference combinations using L0-L1 vs. pairs



The main benefit of the unified list approach is that the selection of reference frame combinations in bi-directional prediction is not restricted to all combinations of reference frames from two lists (as per the current design). Instead, smaller, equal or larger sets of reference frame combinations can be used.

3 Unification with uni-directional prediction

Given a single list of pairs for bi-directional prediction, the concept may be extended to include uni-directional prediction by introducing a “null” reference (also denoted as \emptyset or $-$), so that a pair can describe uni-directional modes, e.g. (\emptyset, a) or (a, \emptyset) . In this way, a single method for describing references is achieved¹.

The full set of possible pairs is illustrated in figure 3, where n denotes any additional reference picture. The set of signalled pairs can include some or all of the pairs containing \emptyset .

4 Implementation

The initial implementation has attempted to preserve the behaviour of the current HM-3.0 reference selection by allowing the exact same set of references to be specified using pairs as would have been using L0 and L1 lists. I.e., no pruning of the pair list has occurred in this emulation mode. In the following the output of the encoder for the first 10 pictures encoded with the random access configuration:

```
POC 0 TId: 0 ... [LU ] [LUP ]
POC 8 TId: 0 ... [LU 0 ] [LUP (0,-) (0,0) ]
POC 4 TId: 0 ... [LU 0 8 ] [LUP (0,-) (-,8) (0,8) (0,0) (8,8) (8,0) ]
POC 2 TId: 0 ... [LU 0 4 8 ] [LUP (0,-) (-,4) (-,8) (0,4) (0,8) (4,4) (4,8) ]
POC 6 TId: 0 ... [LU 4 8 2 ] [LUP (4,-) (-,8) (2,-) (4,8) (4,4) (2,8) (2,4) ]
POC 1 TId: 0 ... [LU 0 2 4 ] [LUP (0,-) (-,2) (-,4) (0,2) (0,4) (2,2) (2,4) ]
POC 3 TId: 0 ... [LU 2 4 0 6 ] [LUP (2,-) (-,4) (0,-) (-,6) (2,4) (2,6) (0,4) (0,6) ]
POC 5 TId: 0 ... [LU 4 6 2 8 ] [LUP (4,-) (-,6) (2,-) (-,8) (4,6) (4,8) (2,6) (2,8) ]
POC 7 TId: 0 ... [LU 6 8 4 ] [LUP (6,-) (-,8) (4,-) (6,8) (6,6) (4,8) (4,6) ]
POC 16 TId: 0 ... [LU 8 6 4 2 ] [LUP (8,-) (6,-) (4,-) (2,-) (8,8) (8,6) (8,4) (8,2)
(6,8) (6,6) (6,4) (6,2) (4,8) (4,6) (4,4) (4,2) (2,8) (2,6) (2,4) (2,2) ]
```

Note that the picture concluding the second GOP, that is POC 16, has four references in each list in the default L0-L1 setting of HM, so in total 16 bi-directional pairs are used.

Subsequent modifications have allowed the configuration of an explicit subset of pairs. An example of a pair configuration as implemented in the software, showing one possible selection of pairs for the case where four references are available (two in each direction) is presented in figure 4. In the case the references 0, 4, 8, n are indexed as 0, 1, 2, 3, the set of enabled pairs as shown would correspond to 4 uni-

¹Due to this property the single list is termed a *unified reference list* (LU)

Figure 3: Complete set of possible pair modes

	$(\emptyset, 0)$	$(\emptyset, 4)$	$(\emptyset, 8)$	\dots	(\emptyset, n)
$(0, \emptyset)$	$(0, 0)$	$(0, 4)$	$(0, 8)$		$(0, n)$
$(4, \emptyset)$	$(4, 0)$	$(4, 4)$	$(4, 8)$		$(4, n)$
$(8, \emptyset)$	$(8, 0)$	$(8, 4)$	$(8, 8)$		$(8, n)$
\vdots				\ddots	\vdots
(n, \emptyset)	$(n, 0)$	$(n, 4)$	$(n, 8)$	\dots	(n, n)

directional $\{(0, \emptyset), (\emptyset, 4), (8, \emptyset), (\emptyset, n)\}$, and 4 bi-directional $\{(0, 4), (0, 8), (4, 4), (8, n)\}$. The order of pairs is determined by numbers 0-7 at the corresponding table entries; the value -1 represents unused pairs.

Figure 4: Example of selection of pairs

```
const int pairsTableRefs2_2[5][5] = {
/* - 0 1 2 3 */
{-1, -1, 1, -1, 3}, // - uni-directional pairs
{0, -1, 4, 5, -1}, // 0
{-1, -1, 6, -1, -1}, // 1
{2, -1, -1, -1, 7}, // 2
{-1, -1, -1, -1, -1}, // 3
}
```

The implementation supports all possible sets of pairs defined with tables similar to figure 4, allowing an arbitrary ordering. Coding of the modes for CAVLC is done as a combination of unary codes and fixed codes. For CABAC unary coding binarisation is used.

5 Syntax and semantics

5.1 Prediction unit

The implemented syntax for the PU is given in table 1. The semantics of `mvd0`, `mvd1`, `mvp_idx0` and `mvp_idx1` are the same as for `mvd_l0`, `mvd_l1`, `mvp_idx_l0` and `mvp_idx_l1` in the current HM draft, respectively.

ref_idx_pair is the index into the reference pairs list

5.2 Slice header

This part has not been implemented fully, only the emulation mode is currently enabled. The explicit mode is currently hard-coded, and while the mode selection is signalled, the list of pairs is not. Table 2 presents a possible syntax.

ref_pic_pair_mode specifies method for deriving pairs to be used in motion compensation.

If `ref_pic_pair_mode` is equal to 0 the pairs are produced as they would be in the usual two-list approach (emulation mode).

If `ref_pic_pair_mode` is equal to 1 the pairs are signalled in the bit-stream (explicit mode).

If `ref_pic_pair_mode` is equal to 2 the pairs are produced by following the process of automatic derivation (automatic mode).

num_ref_idx_active_override_flag equal to 1 specifies that the syntax element `num_ref_idx_mv0_active_minus1` is present for P and B slices and that the syntax element `num_ref_idx_mv1_active_minus1` is present for B slices. `num_ref_idx_active_override_flag` equal to 0 specifies that the syntax elements `num_ref_idx_mv0_active_minus1` and `num_ref_idx_mv1_active_minus1` are not present.

When the current slice is a P or B slice and `field_pic_flag` is equal to 0 and the value of `num_ref_idx_mv0_default_active_minus1` in the picture parameter set exceeds 15, `num_ref_idx_active_override_flag` shall be equal to 1.

When the current slice is a B slice and `field_pic_flag` is equal to 0 and the value of `num_ref_idx_mv1_default_active_minus1` in the picture parameter set exceeds 15, `num_ref_idx_active_override_flag` shall be equal to 1.

Table 1: Modified syntax for PU

prediction_unit() {	C	Descriptor
if(skip_flag[x0][y0]) {		
...		
} else if(PredMode == MODE_INTRA)		
...		
} else { /* MODE_INTER */		
...		
if(merge_flag[x0][y0] && NumMergeCand > 1) {		
merge_idx[x0][y0]		
} else {		
red_idx_pair [x0][y0]		ue(v) ae(v)
if(ref_pic_pair_list[ref_idx_pair[x0][y0]][0] > 0) {		
mvd0 [x0][y0][0]		se(v) ae(v)
mvd0 [x0][y0][1]		se(v) ae(v)
if(NumMVPCand(MV0) > 1)		
mvdp_idx0 [x0][y0]		ue(v) ae(v)
}		
if(ref_pic_pair_list[ref_idx_pair[x0][y0]][1] > 0) {		
mvd1 [x0][y0][0]		se(v) ae(v)
mvd1 [x0][y0][1]		se(v) ae(v)
if(NumMVPCand(MV1) > 1)		
mvdp_idx1 [x0][y0]		ue(v) ae(v)
}		
}		
}		

idx_mv1_default_active_minus1 in the picture parameter set exceeds 15, num_ref_idx_active_override_flag shall be equal to 1.

num_ref_idx_mv0_active_minus1 specifies the number of references from reference picture list that shall be used with MV0 of the motion vector field in the slice. It is used in construction of reference pairs list, in a way defined by the currently selected ref_pic_pair_mode.

num_ref_idx_mv1_active_minus1 exists if current slice_type is B, specifies the number of references from reference picture list that shall be used with MV1 of the motion vector field in the slice. It is used in construction of reference pairs list, in a way defined by the currently selected ref_pic_pair_mode.

num_ref_pair_minus1 specifies the number of reference pairs, if ref_pic_pair_mode is equal to 1 or 2, otherwise this syntax element is not present. num_uni_pair is present when ref_pic_pair_mode is equal to 2 and it specifies the number of reference pairs where one of the elements in a pair does not reference a reference picture (uni-directional cases).

ref_pic_pair_list[n][0] specifies the reference picture index used for n-th element of the reference pairs list, such that the reference picture index is derived as ref_pic_pair_list[n][0] - 1. If it is equal to 0 it specifies that the MV0 is not used.

ref_pic_pair_list[n][1] exists if current slice_type is B, specifies the reference picture index used for n-th element of the reference pairs list, such that the reference picture index is derived as ref_pic_pair_list[n][1] - 1. If equal to 0 it specifies that the MV1 is not used.

ref_pic_list_modification() is a process by which the order of references in the unified list can be modified.

ref_pic_pair_list_modification() is a process by which the order of pairs in the list of pairs can be

Table 2: Modified syntax for slice header

slice_header() {	C	Descriptor
...		
ref_pic_pair_mode		ue(v)
num_ref_idx_active_override_flag		u(1)
if(num_ref_idx_active_override_flag) {		
num_ref_idx_mv0_active_minus1		ue(v)
if(slice_type == B)		
num_ref_idx_mv1_active_minus1		ue(v)
}		
ref_pic_list_modification()		
if(ref_pic_pair_mode == 1 ref_pic_pair_mode == 2)		
num_ref_pair_minus1		ue(v)
if(ref_pic_pair_mode == 2 && slice_type == B)		
num_uni_pair		ue(v)
if(ref_pic_pair_mode == 1) {		
for(n = 0; n <= num_ref_pair_minus1; n++) {		
ref_pic_pair_list[n][0]		u(v)
if(slice_type == B)		
ref_pic_pair_list[n][1]		u(v)
}		
}		
ref_pic_pair_list_modification()		
...		
}		

modified.

Further to this, note that in our proposal sending of a flag `collocated_from_l0_flag` is not necessary as the derivation of motion vector predictors loops over all available references.

5.3 Construction of unified reference list

The unified reference picture list `ref_pic_list` is ordered such that short-term reference entries have lower indices than long-term reference entries. It is ordered as follows:

1. Let `entryShortTerm0` `entryShortTerm1` be variables ranging over all reference entries that are currently marked as “used for short term reference” and which have a value of `temporal_id` equal to or lower than the `temporal_id` of the current picture, and `CurrPic` is the current picture. For each `entryShortTerm0` not equal to `entryShortTerm1` it is defined:

$$\text{entryShortTermDist0} = \text{Abs}(\text{PicOrderCnt}(\text{entryShortTerm0}) - \text{PicOrderCnt}(\text{CurrPic}))$$

$$\text{entryShortTermDist1} = \text{Abs}(\text{PicOrderCnt}(\text{entryShortTerm1}) - \text{PicOrderCnt}(\text{CurrPic}))$$

Then, if `entryShortTermDist0 < entryShortTermDist1`, then `entryShortTerm0` is placed closer to the beginning of `refPicList`. If `entryShortTermDist0 == entryShortTermDist1` then if `PicOrderCnt(entryShortTerm0) < PicOrderCnt(entryShortTerm1)`, then `entryShortTerm0` is placed closer to the beginning of `refPicList`.

NOTE: this sorts short term references in the order of increasing `PicOrderCnt` distance to `CurrPic`, and in the case there are two short-term references with an equal distance, than one with a lower `PicOrderCnt` is placed before the other one.

2. The long-term reference entries which have a value of `temporal_id` equal to or lower than the `temporal_id` of the current picture are ordered starting with the long-term reference entry that has the lowest `LongTermPicNum` value and proceeding through in ascending order to the long-term

reference entry that has the highest LongTermPicNum value.

5.4 Construction of reference pairs list

The reference pair at `ref_idx_pair` position of the reference pairs list `ref_pic_pair_list`, is composed of two elements: `ref_pic_pair_list[ref_idx_pair][0]` and `ref_pic_pair_list[ref_idx_pair][1]`. In the case they are not equal to 0, then `ref_pic_pair_list[ref_idx_pair][0] - 1` and `ref_pic_pair_list[ref_idx_pair][1] - 1` represent the corresponding reference's index into unified reference list `ref_pic_list` for MV0 and MV1, respectively. For slice_type is equal to P, `ref_pic_pair_list[ref_idx_pair][1]` shall be 0. `ref_pic_pair_list[ref_idx_pair][0]` is defined in the range of 0 to `num_ref_idx_mv0_active_minus1 + 1`, while `ref_pic_pair_list[ref_idx_pair][1]` is defined in the range of 0 to `num_ref_idx_mv1_active_minus1 + 1`.

The construction process is different depending on the value of `ref_pic_pair_mode`, which also defines semantics of `num_ref_idx_mv0_active_minus1` and `num_ref_idx_mv1_active_minus1`:

- If `ref_pic_pair_mode` is equal to 0 then `num_ref_idx_mv0_active_minus1 + 1` and `num_ref_idx_mv1_active_minus1 + 1` specify the number of active reference pictures that are used in construction of pairs in this mode, in a way that emulates the two-list (L0 and L1) approach. In that case `num_ref_idx_mv0_active_minus1` and `num_ref_idx_mv1_active_minus1` are equivalent to the `num_ref_idx_l0_active_minus1` syntax element in AVC/H.264.
- If `ref_pic_pair_mode` is equal to 1 or 2 then `num_ref_idx_mv0_active_minus1 + 1` and `num_ref_idx_mv1_active_minus1 + 1` specify the number of active reference pictures used for MV0 and MV1, respectively, which are taken from the front of the unified reference picture list. If `ref_pic_pair_mode` is equal to 1 then it also determines the bit-width of syntax element `ref_pic_pair_list[n][0]`.
- If `ref_pic_pair_mode` is equal to 2 the reference pairs list is derived following some algorithm shared by encoder and decoder. We do not specify such algorithm currently, however, it can follow such criteria as weighted temporal distance of the reference pictures in a pair, direction of reference pictures in a pair, number of non-null entries in a pair (preference to bi-directional or uni-directional modes), etc.

5.5 Picture parameter set

`num_ref_idx_mv0_default_active_minus1` specifies how `num_ref_idx_mv0_active_minus1` is inferred for P and B slices with `num_ref_idx_active_override_flag` equal to 0. The value of `num_ref_idx_mv0_default_active_minus1` shall be in the range of 0 to 31, inclusive.

`num_ref_idx_mv1_default_active_minus1` specifies how `num_ref_idx_mv1_active_minus1` is inferred for B slices with `num_ref_idx_active_override_flag` equal to 0. The value of `num_ref_idx_mv1_default_active_minus1` shall be in the range of 0 to 31, inclusive.

Note also that it would be possible for `ref_pic_pair_list` to be signalled in PPS, and referenced in slice header.

6 Results

The unified reference list has been implemented in HM-3.0. Although the results in this proposal are not presented for the case of explicit or automatic mode of reference pair construction, implementation has been driven by the idea of full support for an arbitrary selection of pairs. The following is the high-level overview of the introduced changes:

- The use of lists L0 and L1 has been removed from most lower-level parts of the codec, and where necessary the notion of lists has been replaced with the notion of motion vector field components, i.e. MV0 and MV1.
- Several functions related to ME process and selection of MVP have been largely rewritten, for instance `TEncSearch::predInterSearch()`, `TEncSearch::xCheckBestMVP()` and `TEncSearch::fillMvpCand()`.
- Some macros have been assumed to have the default setting, for instance `TI_AMVP_SMVP_SIMPLIFIED` and `MTK_AMVP_SMVP_DERIVATION`.

- Combined list has been removed (code within DCM_COMB_LIST macro).
- Construction of the unified reference list and the table of pairs has been done in TComSlice.cpp.

Although the process of selecting the MV candidates for ME refinement and selection of the best MV has been rewritten (in TEncSearch::predInterSearch()), the basic algorithm can be replicated if software is configured without the additional options (as described below). Without the additional options the only difference to basic performance is in the coding of the reference pair index, used in place of coding of inter mode and reference indices. The coding of pair index is described in the following:

- VLC: ref_idx_pair uses a combination of unary and fixed codes. If number of pairs is num_ref_pair_minus1 ≤ 8 then a unary code is used with parameter maxval equal to num_ref_pair_minus1. Otherwise for pair indices lower than 8, a unary code with parameter maxval equal to 8 is used, and pair indices higher or equal to 8 are signalled with a unary code 8 with maxval equal to 8. The remaining num_ref_pair_minus1 - 8 indices are encoded with a fixed number of bits. This encoding scheme has been constructed this way to be similar to the exception case handling in HM-3.0. Further to this, the VLC table adaptation is done for all pairs, not just for first 8.
- CABAC: implementation uses a direct unary code binarisation of the pair index, without context modelling and trained initial states.

The results obtained with this configuration are presented in table 3. Note that these and subsequent results (until the final results) omit the encoder and decoder run-times as the experiment were run under different cluster configuration than anchors. All tests were performed for ra_he, ra_lc, ld_he and ld_lc coding configuration [E700] and the results are compared to HM-3.0 anchors.

Table 3: Results with configuration replicating the HM-3.0 ME and MVP selection process

	Y' BD-rate	U BD-rate	V BD-rate
Class A	0.0	0.0	0.2
Class B	0.1	0.0	0.0
Class C	-0.1	0.0	-0.1
Class D	-0.1	-0.1	-0.1
Class E			
All	0.0	0.0	0.0

(a) Random Access

	Y' BD-rate	U BD-rate	V BD-rate
Class A	-0.2	-0.2	0.0
Class B	-0.1	0.0	0.0
Class C	-0.1	-0.2	-0.1
Class D	-0.2	-0.1	-0.1
Class E			
All	-0.1	-0.1	-0.1

(b) Random Access, LoCo

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.3	0.3
Class C	0.0	0.0	0.0
Class D	-0.1	0.2	0.1
Class E	-0.2	-0.7	-0.3
All	-0.1	0.0	0.0

(c) Low Delay(B)

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.1	0.0	-0.1
Class C	0.0	-0.1	0.0
Class D	0.1	-0.1	-0.1
Class E	0.0	-0.1	0.1
All	0.0	0.0	-0.1

(d) Low Delay(B), LoCo

Further to this, in order to facilitate future experiments with an unconstrained selection of reference pairs, the ME estimation process has been modified so it does not create unfair bias to particular pairs, which can arise from not checking certain combinations of references or not processing MV0 and MV1 equally. For instance, in HM-3.0 randomaccess settings, the case where a current picture has two references, one in L0 and one in L1, two of four bi-directional pairs cannot be selected as the current implementation of the selection of the seed for bi-directional ME effectively disables them.

The following modifications have been implemented and tested:

- Bugfix for the reported ticket 128 is inherently included in the new implementation of TEncSearch::predInterSearch(). Gain for this modification is around 0.1%, as reported on the JCT-VC trac.
- MinD - Seed for bi-directional ME is based on minimum distortion criteria, and is selected to be the minimum cost uni-directional candidate from either L0 or L1 list (and not set per picture as in

HM-3.0).

- **CodeLen** - Codeword length for the current pair is obtained from the entropy coder for CAVLC. For CABAC a predefined estimation replicating the default as in HM-3.0 is used. Gain for this modification is probably around 0.1%.
- **ColMVP** - All references are checked when searching for the collocated MVP (and not just the first reference in a pre-selected list as in HM-3.0).
- **SymMB** - Same number of bits is used when estimating cost for the second partition in 2NxN and Nx2N CUs, as the CABAC implementation does not use contexts.

The modified software includes additional options that were not tested for this report (i.e. they were switched off), which include further modifications aiming to remove the bias between MV0 and MV1. Specifically, the first option performs ME for MV1 component as well, but only if predictor is different than for MV0, and the other option performs ME for all possible pairs (not only for the pairs where one of the references is pre-selected, as in HM-3.0).

In the following the impact of each of these additional modifications is tested, but this time the anchors correspond to the configuration that replicates the default algorithms (i.e. results from table 3).

Table 4 summarises the results for the minimum distortion ME seed modification.

Table 4: Results for MinD modification.

	Y' BD-rate	U BD-rate	V BD-rate		Y' BD-rate	U BD-rate	V BD-rate
Class A	-0.1	-0.1	-0.4	Class A	0.0	0.2	0.4
Class B	-0.1	0.0	0.0	Class B	-0.1	0.1	0.0
Class C	-0.1	-0.1	-0.1	Class C	-0.1	0.0	-0.1
Class D	0.0	-0.1	-0.1	Class D	0.0	0.1	-0.1
Class E				Class E			
All	-0.1	-0.1	-0.2	All	0.0	0.1	0.1

(a) Random Access

	Y' BD-rate	U BD-rate	V BD-rate		Y' BD-rate	U BD-rate	V BD-rate
Class A				Class A			
Class B	0.0	-0.1	-0.3	Class B	0.0	0.1	0.0
Class C	0.0	0.2	-0.1	Class C	0.1	0.2	-0.1
Class D	0.0	0.0	0.1	Class D	0.0	0.4	0.2
Class E	0.0	-0.5	0.0	Class E	0.1	0.3	0.0
All	0.0	0.1	-0.1	All	0.1	0.2	0.0

(c) Low Delay(B)

(b) Random Access, LoCo

(d) Low Delay(B), LoCo

Table 5 summarises the results for modification of the pair index codeword length in ME cost estimation. Since this is done only for CAVLC, only the low complexity results are given.

Table 5: Results for CodeLen modification.

	Y' BD-rate	U BD-rate	V BD-rate		Y' BD-rate	U BD-rate	V BD-rate
Class A	-0.2	-0.3	0.0	Class A	-0.2	-0.2	-0.3
Class B	-0.1	-0.2	-0.2	Class B	-0.3	-0.3	-0.5
Class C	-0.2	-0.2	-0.2	Class C	-0.4	-0.5	-0.7
Class D	-0.2	-0.2	-0.3	Class D	-0.4	0.0	-1.0
Class E				Class E			
All	-0.2	-0.2	-0.2	All	-0.3	-0.3	-0.6

(a) Random Access, LoCo

(b) Low Delay, LoCo

Table 6 summarises the results for modification of checking all references for collocated MVPs. Note that the results include a significant gain for sequence SteamLocomotive, of around 3%, which indicates a potential problem with the either current solution or implementation of deriving the collocated MVPs only from one of the lists. The difference in coding efficiency has been traced down to pictures coded

immediately after the I-pictures, where significantly more intra blocks were selected in HM-3.0 than with the ColMVP modification.

Table 6: Results for ColMVP modification.

	Y' BD-rate	U BD-rate	V BD-rate
Class A	−0.8	−1.2	−2.2
Class B	−0.1	−0.1	−0.1
Class C	−0.1	−0.1	0.0
Class D	−0.1	−0.1	−0.2
Class E			
All	−0.3	−0.4	−0.6

(a) Random Access

	Y' BD-rate	U BD-rate	V BD-rate
Class A	−0.9	−0.9	−1.2
Class B	−0.1	−0.1	−0.1
Class C	−0.1	0.0	0.0
Class D	−0.1	0.0	0.0
Class E			
All	−0.3	−0.2	−0.3

(b) Random Access, LoCo

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.0	0.1
Class C	0.0	0.1	0.1
Class D	0.0	−0.3	−0.2
Class E	0.0	0.7	0.3
All	0.0	0.0	0.1

(c) Low Delay(B)

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.0	0.0
Class C	0.0	0.0	0.0
Class D	0.0	0.4	0.1
Class E	0.1	0.3	−0.1
All	0.0	0.2	0.0

(d) Low Delay(B), LoCo

Table 7 summarises the results for using the same number of bits when estimating cost for the second partition in 2NxN and Nx2N CUs .

Table 7: Results for SymMB modification.

	Y' BD-rate	U BD-rate	V BD-rate
Class A	0.0	0.0	−0.3
Class B	0.0	0.0	0.0
Class C	0.0	0.0	0.1
Class D	0.0	0.0	0.0
Class E			
All	0.0	0.0	−0.1

(a) Random Access

	Y' BD-rate	U BD-rate	V BD-rate
Class A	0.0	−0.1	0.1
Class B	0.0	0.0	0.0
Class C	0.0	0.1	0.1
Class D	0.0	0.0	0.1
Class E			
All	0.0	0.0	0.0

(b) Random Access, LoCo

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.0	0.0
Class C	0.0	0.0	0.0
Class D	0.0	0.0	0.0
Class E	0.0	0.0	0.0
All	0.0	0.0	0.0

(c) Low Delay(B)

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.0	0.0
Class C	0.0	0.0	0.0
Class D	0.0	0.0	0.0
Class E	0.0	0.0	0.0
All	0.0	0.0	0.0

(d) Low Delay(B), LoCo

All modifications have been tested for ra_he, ra_lc, ld_he, ld_lc, ldP_he and ldP_lc coding configuration and compared with the HM-3.0 anchors, of which the results are presented in table 8.

7 Extensions and conclusions

The gain reported can be contributed to two factors, namely, (1) a simpler design for joint coding of reference indices and the inter prediction mode, and (2) modifications in the ME process. The said modifications strive to remove bias to a particular prediction direction or reference picture, so the interaction of reference picture selection and motion vector predictors can be studied. Therefore the presented modifications in the software form a basis for experimentation with different picture prediction structures in order to find optimal sets of prediction directions. The required syntax modifications bring simplification to the PU level processes, brought by preserving the joint representation of the reference picture selection and the inter prediction direction throughout operation of the codec, and by removing the requirement to translate between different lists.

Table 8: Results of all modifications

	Y' BD-rate	U BD-rate	V BD-rate
Class A	−1.0	−1.6	−1.9
Class B	−0.2	−0.2	−0.1
Class C	−0.3	−0.2	−0.2
Class D	−0.2	−0.2	−0.1
Class E			
All	−0.4	−0.5	−0.6
Enc Time	101%		
Dec Time	90%		

(a) Random Access

	Y' BD-rate	U BD-rate	V BD-rate
Class A	−1.2	−1.0	−1.2
Class B	−0.3	−0.2	−0.3
Class C	−0.4	−0.5	−0.5
Class D	−0.4	−0.3	−0.4
Class E			
All	−0.6	−0.5	−0.6
Enc Time	101%		
Dec Time	93%		

(b) Random Access, LoCo

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.1	0.2	0.0
Class C	0.0	0.1	0.0
Class D	−0.1	−0.2	0.0
Class E	−0.1	0.3	0.1
All	0.0	0.1	0.0
Enc Time	101%		
Dec Time	91%		

(c) Low Delay(B)

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	−0.1	−0.3
Class C	−0.2	−0.3	−0.5
Class D	−0.3	−0.4	−0.6
Class E	−0.5	−0.7	−0.8
All	−0.2	−0.4	−0.5
Enc Time	102%		
Dec Time	94%		

(d) Low Delay(B), LoCo

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	0.0	0.3	−0.1
Class C	0.0	0.1	0.0
Class D	0.0	−0.2	−0.1
Class E	0.1	0.5	−0.5
All	0.0	0.2	−0.2
Enc Time	101%		
Dec Time	91%		

(e) Low Delay(P)

	Y' BD-rate	U BD-rate	V BD-rate
Class A			
Class B	−0.1	−0.1	−0.1
Class C	−0.1	−0.2	−0.3
Class D	−0.2	−0.7	−0.8
Class E	−0.1	−0.5	−0.2
All	−0.1	−0.4	−0.4
Enc Time	102%		
Dec Time	89%		

(f) Low Delay(P), LoCo

8 Patent rights declaration

The BBC 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).