|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  10th Meeting: Stockholm, Sweden, 11– 20 July 2012 | Document: JCTVC-J0170\_r1 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **On Temporal and Combined motion vector predictors derivation for Merge/Skip mode** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Guillaume Laroche  Tangi Poirier  Patrice Onno  Canon Research Centre France Rue de la Touche Lambert  35510 CESSON-SEVIGNE, FRANCE | Tel: Email: | +33(0)299876800 [guillaume.laroche@crf.canon.fr](mailto:guillaume.laroche@crf.canon.fr)  [patrice.onno@crf.canon.fr](mailto:patrice.onno@crf.canon.fr) |
| *Source:* | Canon Research Centre France | | |

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

# Abstract

This contribution presents a simplification of the motion vector derivation process for the Merge/Skip modes. The aim is to reduce the number of cycles for the worst case for hardware implementation that are needed to perform the whole motion vector prediction derivation. The proposed simplification consists in scaling the temporal candidates in parallel to the derivation of the combined predictors. The proposed modification reports only 0.1% BDR loss.

# Introduction

The derivation of the Merge/Skip candidates has been simplified by reducing the amount of operations in the pruning process and by removing candidates by previous adoptions. But the current version can be still improved to reduce the number of cycles for the worst case regarding the hardware implementation as we proposed in [1].

Figure 1 shows the current derivation of the Merge/Skip candidates. In the first step, the availability of spatial and temporal candidates is checked. Then a simple pruning process is applied in order to remove some of the redundant spatial predictors. For the temporal motion candidate, a scaling is applied if needed. Several contributions analysed the complexity of this scaling process for hardware implementation [2], [3]. When all candidates (spatial and temporal) are available, the combined candidates are generated. In this process, up to 12 combinations of spatial and temporal candidates are created and tested if the motion information of List 0 is not equal to the motion information of List 1. This needs 12 comparisons for the worst case.

In this contribution, we propose to reduce the worst case complexity by cascading the scaling process of the temporal candidate and the combined list generation.



Figure 1 Merge/Skip candidates derivation process in HM7.0.

# Proposed scheme

Figure 2 shows the proposed derivation of the Merge/Skip candidates. The modification consists in disabling the temporal candidate as an input candidate for the generation process of the combined predictors. Consequently, the scaling of the temporal predictor can be processed in parallel to the combined list generation. For non-parallel implementation, the proposed modification reduces the number of operations by avoiding the use of the temporal motion vector in the combined MV candidates. This improves the throughput for the worst case.

The proposed modification is very simple and needs only 3 lines of code modification of the current HM7.0 and minor change of DIS text.

# 

Figure 2 Proposed Merge/Skip candidates derivation process

# Experimental results

The experiments were conducted in accordance with the common tests conditions [4].

Table 1 provides the results of the proposed scheme. In average, for the 6 Inter configurations the proposed scheme gives 0.1% BDR loss. Please note that the combined predictors are not used for P frames. So the low delay P configurations are not affected by the proposed modifications.

Table 1 Simulation results of the proposed simplification.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Random Access Main** | | | **Random Access HE10** | | |
|  | Y | U | V | Y | U | V |
| Class A | 0.1% | -0.1% | 0.0% | 0.1% | -0.1% | 0.2% |
| Class B | 0.1% | 0.1% | 0.1% | 0.1% | 0.0% | 0.0% |
| Class C | 0.1% | 0.2% | 0.2% | 0.1% | 0.1% | 0.0% |
| Class D | 0.1% | 0.1% | 0.1% | 0.1% | 0.1% | 0.1% |
| Class E |  |  |  |  |  |  |
| Class F | 0.0% | 0.0% | 0.1% | 0.0% | 0.0% | 0.1% |
| **Overall** | 0.1% | 0.1% | 0.1% | 0.1% | 0.0% | 0.1% |
|  | 0.1% | 0.1% | 0.1% | 0.1% | 0.0% | 0.1% |
| Enc Time[%] | 100% | | | 101% | | |
| Dec Time[%] | 100% | | | 101% | | |
|  |  |  |  |  |  |  |
|  | **Low delay B Main** | | | **Low delay B HE10** | | |
|  | Y | U | V | Y | U | V |
| Class A |  |  |  |  |  |  |
| Class B | 0.2% | 0.2% | 0.0% | 0.2% | 0.2% | 0.2% |
| Class C | 0.2% | 0.2% | 0.2% | 0.2% | 0.2% | 0.4% |
| Class D | 0.1% | 0.1% | 0.4% | 0.2% | 0.5% | 0.2% |
| Class E | -0.1% | 0.3% | -0.1% | 0.1% | -0.2% | -0.2% |
| Class F | -0.1% | 0.3% | 0.2% | 0.2% | 0.4% | 1.0% |
| **Overall** | 0.1% | 0.2% | 0.1% | 0.2% | 0.2% | 0.3% |
|  | 0.1% | 0.2% | 0.1% | 0.2% | 0.3% | 0.4% |
| Enc Time[%] | 99% | | | 100% | | |
| Dec Time[%] | 100% | | | 99% | | |
|  |  |  |  |  |  |  |

# Conclusions

This contribution presents a simplification of the derivation of the Merge/Skip candidates in order to increase the throughput of the current HEVC design. The proposed modification consists in disabling the use of the temporal predictor to generate the combined candidates. Consequently, the combined list and the temporal scaling can be derived in parallel. The proposed modification gives a small loss of 0.1% in average. This modification has been implemented with only 3 lines modification in the HM source code and needs minor modifications of the current DIS.

# DIS text Modification

##### Derivation process for combined bi-predictive merging candidates

Inputs of this process are

* a merging candidate list mergeCandList,
* reference indices refIdxL0N and refIdxL1N of every candidate N being in mergeCandList,
* prediction list utilization flags predFlagL0N and predFlagL1N of every candidate N being in mergeCandList,
* motion vectors mvL0N and mvL1N of every candidate N being in mergeCandList,
* the number of elements numMergeCand within mergeCandList,
* the number of elements numOrigMergeCand within the mergeCandList after the spatial and temporal merge candidate derivation process,
* the variable availableFlagCol

Outputs of this process are

* the merging candidate list mergeCandList,
* the number of elements numMergeCand within mergeCandList.
* reference indices refIdxL0combCandk and refIdxL1combCandk of every new candidate combCandk being added in mergeCandList during the invokation of this process,
* prediction list utilization flags predFlagL0combCandk and predFlagL1combCandk of every new candidate combCandk being added in mergeCandList during the invokation of this process,
* motion vectors mvL0combCandk and mvL1combCandk of every new candidate combCandk being added in mergeCandList during the invokation of this process,

When (numOrigMergeCand – availableFlagCol) is greater than 1 and when numOrigMergeCand is less than MaxNumMergeCand, the variable numInputMergeCand is set to numMergeCand, the variables combIdx and combCnt are set to 0, the variable combStop is set to FALSE and the following steps are repeated until combStop is equal to TRUE.

1. The variables l0CandIdx and l1CandIdx are derived using combIdx as specified in Table 8‑8.
2. The following assignments are made with l0Cand being the candidate at position l0CandIdx and l1Cand being the candidate at position l1CandIdx in the merging candidate list mergeCandList ( l0Cand = mergeCandList[ l0CandIdx ] , l1Cand = mergeCandList[ l1CandIdx ] ).
3. When all of the following conditions are true,
   * + predFlagL0l0Cand = = 1
     + predFlagL1l1Cand = = 1
     + PicOrderCnt( RefPicList0[ refIdxL0l0Cand ] ) != PicOrderCnt( RefPicList1[ refIdxL1l1Cand ] ) | | mvL0l0Cand != mvL1l1Cand

the following applies.

* + - The candidate combCandk with k equal to ( numMergeCand − numInputMergeCand ) is added at the end of mergeCandList ( mergeCandList[ numMergeCand ] = combCandk ) and the reference indices, the prediction list utilization flags and the motion vectors of combCandk are dervied as follows and numMergeCand is incremented by 1.

refIdxL0combCandk = refIdxL0l0Cand (8-96)

refIdxL1combCandk = refIdxL1l1Cand (8-97)

predFlagL0combCandk = 1 (8-98)

predFlagL1combCandk = 1 (8-99)

mvL0combCandk[ 0 ] = mvL0l0Cand[ 0 ] (8-100)

mvL0combCandk[ 1 ] = mvL0l0Cand[ 1 ] (8-101)

mvL1combCandk[ 0 ] = mvL1l1Cand[ 0 ] (8-102)

mvL1combCandk[ 1 ] = mvL1l1Cand[ 1 ] (8-103)

numMergeCand = numMergeCand + 1 (8-104)

* + - The variable combCnt is incremented by 1.

1. The variable combIdx is incremented by 1.
2. When combIdx is equal to ((numOrigMergeCand – availableFlagCol) \* ( numOrigMergeCand − availableFlagCol − 1 ) ) or numMergeCand is equal to MaxNumMergeCand or combCnt is equal to 5, combStop is set to TRUE.

# References

1. G. Laroche, T. Poirier, P. Onno, “Non-CE9: On parallel derivation of the temporal predictor for Merge/Skip modes”, Document of Joint Collaborative Team on Video Coding, JCTVC-G241, November 2011.
2. M. Zhou, V. Sze, Y. Matsuba, “A study on HM3.0 parsing throughput issue”, Document of Joint Collaborative Team on Video Coding, JCTVC-F068, July, 2011.
3. T.-D. Chuang, J.-L. Lin, C.-Y. Chen, Y.-Wen, Y.-W. Huang, S. Lei, “Non-CE9: Division-free MV scaling”,Document of Joint Collaborative Team on Video Coding, JCTVC-G223, November 2011.
4. F. Bossen, “Common HM test conditions and software reference configurations,” Document of Joint Collaborative Team on Video Coding, JCTVC-I1100, April 2012.

# Patent rights declaration(s)

**Canon Research Centre France 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).**