

3D-CE5.h: Additional merge candidates derived from shifted disparity candidate predictors

Laurent Guillo, Christine Guillemot

► **To cite this version:**

Laurent Guillo, Christine Guillemot. 3D-CE5.h: Additional merge candidates derived from shifted disparity candidate predictors. Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC.. 2013. <hal-00917123>

HAL Id: hal-00917123

<https://hal.inria.fr/hal-00917123>

Submitted on 11 Dec 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Title: **3D-CE5.h: Additional merge candidates derived from shifted disparity candidate predictors**

Status: Input Document

Purpose: Proposal

*Author(s) or
Contact(s):* Christine Guillemot
 Laurent Guillo

Tel: +33 299847100
Email: Christine.guillemot@inria.fr
 Laurent.guillo@inria.fr

Source: INRIA, France.

Abstract

The coding performance of the HTM 5.0.1 can be improved by inserting in the candidate merge list two new candidates, the maximal length of list still being equal to 6. These candidates are derived from the first disparity compensated candidate found in the list, which is then horizontally shifted by +4 and -4. Bit rate gains are respectively 0.4% and 0.5% for the side views 1 and 2. The overall bit rate gain is 0.2%.

1 Introduction

In 3D-HEVC, up to six merging candidates are made available for selection when inter-view motion prediction is enabled. The efficiency of this approach relies on how relevant vectors present in the list are. In HTM 5.0.1, several adoptions improved the candidate merge list in pruning duplicated candidates and in inserting, when IMPV is available, disparity motion vector derived from disparity vector [1][2].

It is still possible to improve the coding performance by adding two new candidates derived from the first disparity compensated candidate found in the candidate merge list in adding a horizontal shift of -4 or +4 to it. That is equivalent to a horizontal shift of -1 or +1 pixel.

The method described below proposes to replace existing candidates of the candidate merge list described in the text specification draft with two candidates pointing to the base view and horizontally shifted by the offset -4 and +4 respectively.

This proposal is the follow-up of JCT3V-B0080.

2 Proposed method

First, the merge candidate list is constructed as described in the section “G.8.5.2.1.1 Derivation process for luma motion vectors for merge mode” [3]. Nothing is changed with respect to the way it is done in HTM 5.0.1 A possible candidate merge list could be as depicted in Figure 1:

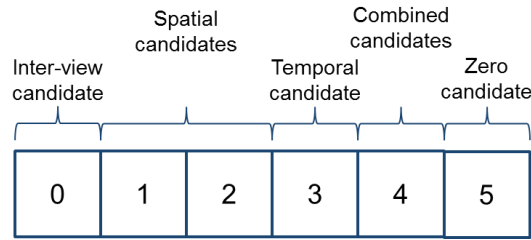


Figure 1: Example of a candidate merge list.

In this list, each motion vector candidate is defined with:

- The luma motion vectors $mvL0$ and $mvL1$,
- The reference indices $refIdxL0$ and $refIdxL1$,
- The prediction list utilization flags $predFlagL0$ and $predFlagL1$.

Once the candidate merge list is constructed, the refinement of the list is a three step process:

1. Search of the 1st disparity compensated predictor,
2. Creation of the two new candidates,
3. Insertion of the new candidates into the candidate merge list.

These three steps are detailed below.

2.1 Search of the 1st disparity compensated predictor

The candidate merge list is scanned from the first position (index 0) to the third one (index 2). The first candidate found in the list during this search which has either its L0 vector or L1 vector or both pointing to another view is marked as *selected candidate* (a vector $mvLX$ is pointing to another view if the $viewId$ of current coding unit is not equal to the $viewId$ of reference $refIdxLX$).

If no such candidate exists, the process ends.

2.2 Creation of the two new candidates

Let m be the index of the *selected candidate*. This index m is higher or equal to 0 and lower or equal to 2. The selected candidate is defined by $mvL0_m$, $mvL1_m$, $refIdxL0_m$, $refIdxL1_m$, $predFlagL0_m$, and $predFlagL1_m$. Two refined candidates $candDCPa$ and $candDCPb$ are constructed as follows:

```

If  $mvL0_m$  is pointing to another view:
 $mvL0candDCPa = mvL0_m + (4, 0)$ 
 $mvL1candDCPa = mvL1_m$ 
 $refIdxL0candDCPa = refIdxL0_m$ 
 $refIdxL1candDCPa = refIdxL1_m$ 
 $predFlagL0candDCPa = predFlagL0_m$ 
 $predFlagL1candDCPa = predFlagL1_m$ 

 $mvL0candDCPb = mvL0_m + (-4, 0)$ 
 $mvL1candDCPb = mvL1_m$ 

```

```

refIdxL0candDCPb = refIdxL0m
refIdxL1candDCPb = refIdxL1m
predFlagL0candDCPb = predFlagL0m
predFlagL1candDCPb = predFlagL1m

```

Else (mvL1_m vector is pointing to another view):

```

mvL0candDCPa = mvL0m
mvL1candDCPa = mvL1m + (4, 0)
refIdxL0candDCPa = refIdxL0m
refIdxL1candDCPa = refIdxL1m
predFlagL0candDCPa = predFlagL0m
predFlagL1candDCPa = predFlagL1m

```

```

mvL0candDCPb = mvL0m
mvL1candDCPb = mvL1m + (-4, 0)
refIdxL0candDCPb = refIdxL0m
refIdxL1candDCPb = refIdxL1m
predFlagL0candDCPb = predFlagL0m
predFlagL1candDCPb = predFlagL1m

```

2.3 Insertion of the new candidates into the candidate merge list

The two new candidates will be added at the 4th position (index 3) and 5th position (index 4). The previous 4th and 5th candidates might be overwritten. However, the previous 4th candidate will be moved to the 6th position (index 5) first. These two operations are depicted in the following figures and the pseudo code listed after.

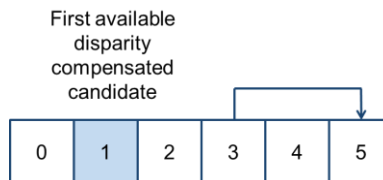


Figure 2: Moving of the fourth candidate to the last position in the candidate list.

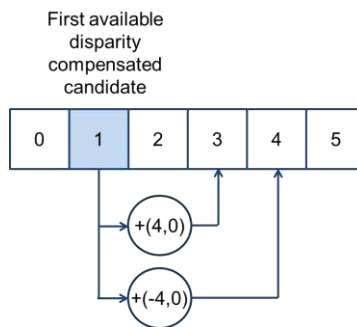


Figure 3: Overwriting of fourth and fifth candidates by newly generated candidates.

```
// insertion of the two new candidates
```

```

mvL05 = mvL03
mvL15 = mvL13
refIdxL05 = refIdxL03
refIdxL15 = refIdxL13
predFlagL05 = predFlagL03
predFlagL15 = predFlagL13

mvL03 = mvL0candDCPa
mvL13 = mvL1candDCPa
refIdxL03 = refIdxL0candDCPa
refIdxL13 = refIdxL1candDCPa
predFlagL03 = predFlagL0candDCPa
predFlagL13 = predFlagL1candDCPa

mvL04 = mvL0candDCPb
mvL14 = mvL1candDCPb
refIdxL04 = refIdxL0candDCPb
refIdxL14 = refIdxL1candDCPb
predFlagL04 = predFlagL0candDCPb
predFlagL14 = predFlagL1candDCPb

```

3 Experimental results

This section provide results of the proposed method in comparison with the 3D-HTM anchor. The 3D-HTM 5.0.1 [4] is utilized and the method is implemented on top of it. All tests are run based on the common tests conditions [5].

Only the method `TComDataCU::getInterMergeCandidates` in `TComDataCU.cpp` is modified (and also `TypeDef.h` to activate the macro `INRIA_NEW_MERGE_CAND`).

Results are provided in table 1. The overall average bitrate reduction is 0.2%, 0.1%, 0.2% for decoded texture views, synthesized views, coded and synthesized views, respectively.

No added complexity is reported.

Table 1: Coding gains with respect to anchor for 3-view case

	video 0	video 1	video 2	video only	synthesized only	coded & synthesized	enc time	dec time	ren time
Balloons	0,0%	0,2%	0,0%	0,0%	0,0%	0,0%	95,5%	102,0%	106,3%
Kendo	0,0%	-0,4%	-0,7%	-0,3%	-0,1%	-0,1%	97,4%	101,9%	110,7%
Newspapercc	0,0%	-0,7%	-0,6%	-0,3%	-0,2%	-0,2%	103,3%	98,2%	113,8%
GhostTownFly	0,0%	-0,5%	-0,9%	-0,2%	-0,2%	-0,2%	101,6%	102,4%	106,0%
PoznanHall2	0,0%	-0,2%	-0,2%	-0,1%	-0,1%	-0,1%	99,4%	99,0%	104,4%
PoznanStreet	0,0%	-0,7%	-0,6%	-0,3%	-0,2%	-0,2%	102,9%	101,1%	106,7%
UndoDancer	0,0%	-0,3%	-0,4%	-0,1%	-0,2%	-0,2%	101,6%	101,5%	104,8%
1024x768	0,0%	-0,3%	-0,4%	-0,2%	-0,1%	-0,1%	98,7%	100,7%	110,2%
1920x1088	0,0%	-0,4%	-0,5%	-0,2%	-0,2%	-0,2%	101,4%	101,0%	105,5%
average	0,0%	-0,4%	-0,5%	-0,2%	-0,1%	-0,2%	100,2%	100,9%	107,5%

4 Conclusion

This proposal provides an overall bit rate gain of 0.2% with no added complexity. It is proposed to adopt it.

5 References

- [1] L. Zhang, Y. Chen, "3D-CE5.h: Merge candidates derivation from disparity vector", ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, JCT3V-B0048, 2nd Meeting: Shanghai, CN, 13–19 Oct 2012.
- [2] E. Mora, B. Pesquet-Popescu, M. Cagnazzo, J. Jung, "3D-CE5.h related: Modification of the Merge Candidate List for Dependant Views in 3DV-HTM", ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, JCT3V-B0069, 2nd Meeting: Shanghai, CN, 13–19 Oct 2012.
- [3] G. Tech, K. Wegner, Y. Chen, S. Yea, "3D-HEVC Test Model 2", ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, JCT3V-B1005, 2nd Meeting: Shanghai, CN, 13–19 Oct 2012.
- [4] 3DV-HTM version 5.0.1: https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSsoftware/tags/HTM-5.0.1/.
- [5] D. Rusanovskyy, K. Müller, A. Vetro, "Common Test Conditions of 3DV Core Experiments," ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, JCT3V-B1110, 2nd Meeting: Shanghai, CN, 13–19 Oct 2012.

6 Patent rights declaration(s)

INRIA 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).