

REDEFINING MOBILITY



JCT3V-D0185: MB-level NBDV for 3D-AVC

Ying Chen, Jewon Kang, Li Zhang, Ye-Kui Wang, Xin Zhao, Rajan Joshi, and Marta Karczewicz

JCT3V-D0186: Derived disparity vector in 3D-AVC

Xin Zhao, Ying Chen, Li Zhang and Marta Karczewicz

Background

- Both proposals are to address the USNB comment on multiview compatibility of 3D-AVC
- Two separate proposals are made
 - The first proposal is identical to the technical solution shown in USNB and therefore well known by several participant companies
- The reason two separate proposals are presented together is
 - They are for the same problem
 - The second proposal is a very simple improvement of the first one

Abstract

- This input document addresses the USNB comment on multiview compatibility of 3D-AVC.
- The current 3D-AVC codec supports flexible decoding order for depth and texture. However, there is a big performance gap between the following 2 configurations:
 - Config. 1 (aka depth-first coding): texture followed by depth for the base view, depth followed by texture for other views.
 - Config. 2 (aka texture-first coding): texture followed by depth for all views.

Abstract

- Coding performance

- Config. 1 outperforms config. 2 by about 19%.
 - Config. 2 provides only negligible coding gain (about 1%) for texture views compared to MVC+D.
-
- Since the decoding order in config. 2 is needed for stereo/multiview compatibility, it is important that the coding of configuration 2 be improved to achieve significantly higher coding efficiency for texture views as compared to MVC+D.

Abstract

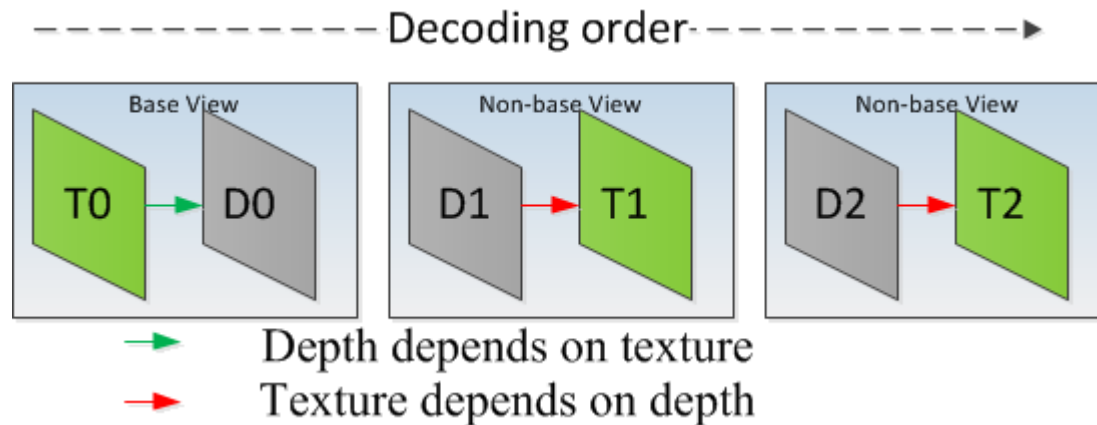
- In addition, we provide a description of a technical solution, which is shown to be able to improve the performance of the texture-first coding mode with comparable (actually slightly better) performance as the depth-first coding mode (the current best performing mode of 3D-AVC).
- The technical solution consists of introduction of the so-called Neighboring Block based Disparity Vector (NBDV), to be applied in addition to **all existing coding tools** that bring coding gains for the depth-first coding mode but not the texture-first coding mode.
 - NBDV is included in 3D-HEVC, and the related text is quite mature.
 - It is proposed to simplify and re-use NBDV in 3D-AVC
- Specification change:
 - Only one page of text needs to be added.
 - In addition moderate amount of text needs to be deleted or slightly changed.
- The change would not be so significant as to require a delay of the schedule of this project.

Introduction

- Flexible coding order in 3D-AVC
 - Depth-first coding
 - Texture-first coding
- Coding efficiency results of the current 3D-AVC
 - 3D-AVC (depth-first coding) vs MVC+D
 - 3D-AVC (texture-first coding) vs MVC+D
 - 3D-AVC texture-first coding vs 3D-AVC depth-first coding

Depth-first coding

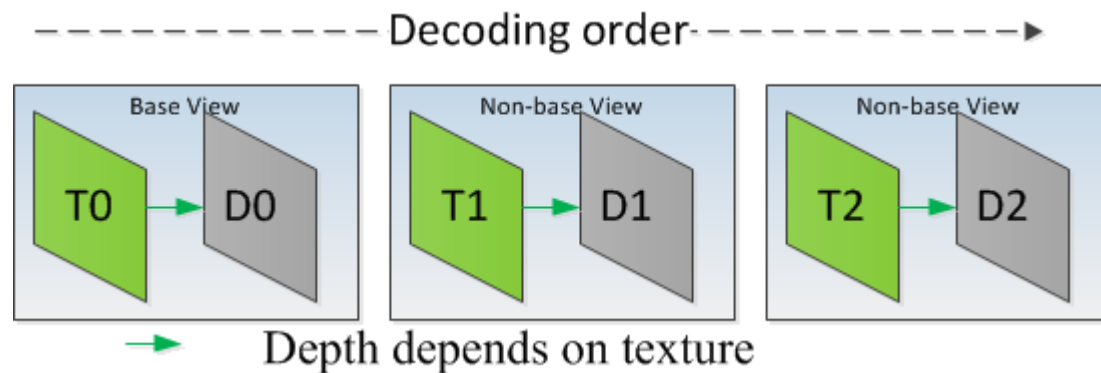
- In such a configuration, to decode the texture view component of each non-base view, its associated depth view component (of the same view) needs also to be decoded.



- The main motivation of depth-first coding
 - To facilitate the disparity vector generation for each non-base view such that inter-view prediction coding tools can be made more efficient.
- However it is questionable such a configuration is really needed if there is a method of deriving disparity vectors when just accessing the texture views.

Texture-first coding

- In such a configuration, to decode the texture view component of each non-base view, its associated depth view component (of the same view) needs also to be decoded.



- Texture-first coding is enabled in the current 3D-AVC codec to meet an MPEG requirement on stereo/multiview compatibility.
 - As stated in 3DV CfP: “The compressed data format shall include a mode that enables the simple extraction of bitstreams for stereo and mono output, and support high-fidelity reconstruction of samples from the left and right views of the stereo video.”

Depth-first coding performance

- Used as common test condition
- Best performing configuration of 3D-AVC

3D-AVC (depth-first coding) vs MVC+D

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	41.74	-1.12	-7.08	0.38	46.13	-1.23	43.67	-1.20
S02	13.10	-0.36	-19.68	0.92	16.39	-0.45	14.78	-0.42
S03	22.77	-0.71	-25.91	2.39	20.94	-0.67	23.87	-0.69
S04	30.08	-0.96	4.28	-0.23	30.39	-0.98	29.92	-0.91
S05	30.47	-1.28	-13.56	0.73	42.84	-1.66	39.45	-1.42
S06	37.77	-1.63	-14.95	0.73	44.39	-1.82	38.38	-1.49
S08	17.33	-0.69	-24.86	1.21	30.80	-1.14	22.00	-0.72
Average	27.61	-0.96	-14.54	0.87	33.13	-1.14	30.29	-0.98

Texture-first coding performance

- The texture-first coding mode of 3D-AVC provides almost the same (1% better) coding performance as MVC+D considering the texture only operation point.

3D-AVC (texture-first coding) vs MVC+D

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	0.54	-0.02	-4.05	0.21	6.60	-0.24	9.12	-0.33
S02	0.34	-0.01	-15.08	0.67	4.85	-0.15	5.20	-0.17
S03	0.40	-0.01	-20.24	1.83	0.51	-0.02	7.17	-0.23
S04	0.11	0.00	19.75	-1.05	2.76	-0.11	6.29	-0.23
S05	2.16	-0.12	-11.85	0.63	16.59	-0.79	15.43	-0.68
S06	3.14	-0.17	-9.21	0.44	12.05	-0.60	10.44	-0.48
S08	1.93	-0.09	-22.30	1.07	15.81	-0.65	9.60	-0.35
Average	1.23	-0.06	-9.00	0.54	8.45	-0.37	9.03	-0.35

Comparison of two configurations

- It is noticed that texture-first coding has a significant loss, by 19%, compared to depth-first coding, as configured in the common test condition (CTC) of JCT-3V.

3D-AVC texture-first coding vs 3D-AVC depth-first coding

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	40.72	-1.09	-2.87	0.19	36.69	-1.01	31.27	-0.89
S02	12.64	-0.35	-6.49	0.30	10.94	-0.31	9.08	-0.26
S03	22.24	-0.70	-6.58	0.56	20.33	-0.66	15.45	-0.45
S04	29.93	-0.96	-12.22	0.80	26.94	-0.88	22.00	-0.69
S05	27.57	-1.18	-2.88	0.14	22.35	-0.93	20.78	-0.79
S06	33.30	-1.47	-7.22	0.34	28.58	-1.25	25.26	-1.03
S08	14.97	-0.60	-3.88	0.19	12.80	-0.51	11.43	-0.39
Average	25.91	-0.91	-6.02	0.36	22.66	-0.79	19.33	-0.64

Problems

- The best performing configuration (depth-first coding) always requires decoding of the depth view component of a non-base view to decode the texture view component of the same view.
 - It is impossible to have a texture-only operation point.
 - In applications where a texture only operation point is desired, the bandwidth and the decoding complexity are still high, as transmission and decoding of the depth view components are always needed.
 - It requires decoding of six view components for the 3-view case, instead of three view components for MVC+D.
 - For example, when the bitrates of the depth views are taken into account, the depth-first coding configuration outperforms MVC+D for texture coding by around only 12%.

Problems

- When configured in texture-first decoding, so that texture view component can be decoded without accessing its depth view component, i.e, majority of the coding tools cannot work properly due to the missing of disparity vectors.
 - The performance if evaluated by texture views, is about 1% better compared to MVC+D, or MVC.
 - 19% coding loss compared to the common test condition (CTC) with depth-first coding is observed.

Proposal

- An additional tool called Neighboring Block based Disparity Vector (NBDV) is proposed. This is a simplified version of a similar tool, included in 3D-HEVC.
- By utilizing the disparity vector derived from the NBDV, there is no need to access the depth view component of the same view for decoding any texture view component to get disparity vectors converted from depth values.
- The proposed method enables all the existing tools.

Proposal

- Advantages of the proposed method
 - When decoding of a texture view component doesn't depend on any depth view component, the coding performance gap compared to MVC+D is greatly increased
 - from 1% to **22%** for texture views
 - the overall coding efficiency as evaluated by synthesized views increases from 10% to **27%**.
 - This coding performance can be achieved in a stereo and multi-view/stereo compatible fashion with the existing coding tools.
 - In addition, when accessing depth view components of the base view is allowed, thereby enabling coding tools such as BVSP (block-based view synthesis prediction),
 - An additional **4~6%** gain can be achieved,
 - The performance of texture-first coding becomes slightly better than depth-first coding, which is the best performing configuration in the current 3D-AVC design
 - Decoding of a depth view component in a non-base view is not needed anymore for decoding the texture view component.

Proposal

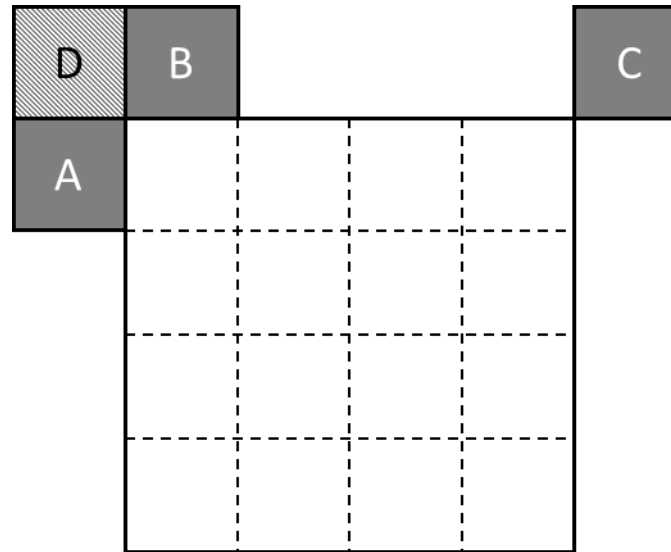
- Summary of the proposed method
 - We propose to introduce MB-level NBDV into 3D-AVC.
 - Similar to 3D-HEVC, neighboring blocks are checked in order.
 - Once a neighboring block contains a disparity motion vector (thus it is inter-view predicted), the motion vector is identified to be a disparity vector, and the whole process terminates.
 - The identified disparity vector is the result of the NBDV process.
 - If a disparity vector is not identified from the neighbors, disparity vector is set to the DDV (JCT3V-D0186)
 - After the disparity vector is derived, DDV is set to the current disparity vector (JCT3V-D0186)

Red text above is the additional part in JCT3V-D0186

- The result of the NBDV process is used to replace the disparity vector which is currently calculated based on the depth view component of the same view under the current 3D-AVC depth-first configuration.
- *Detailed spec. text changes are included and the NBDV process described below is mainly included in subclause J.8.3.1.10.*

Proposal

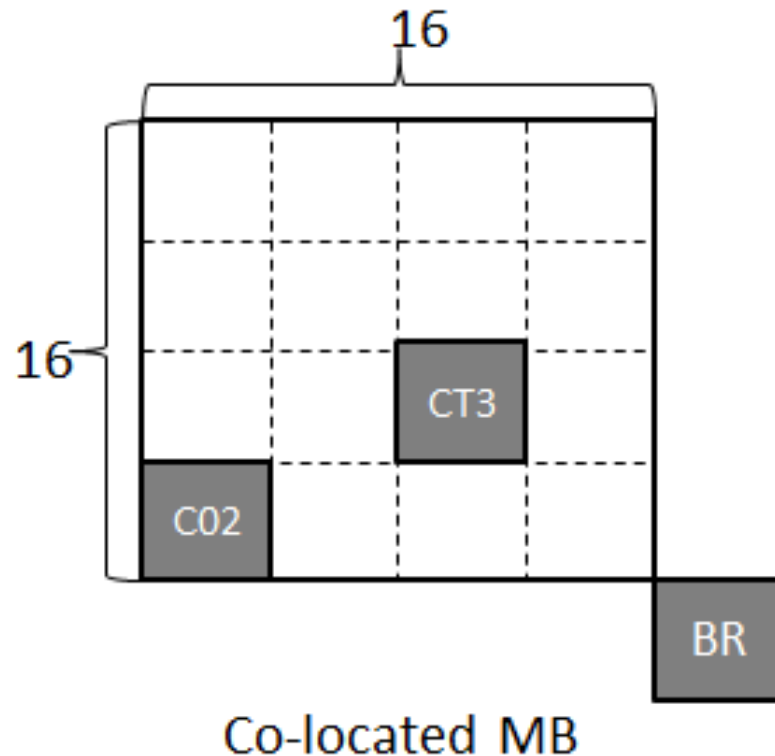
- Spatial neighboring blocks
 - In the order of A, B, C and D.



Currently coded MB

Proposal

- Temporal neighboring blocks
 - In two reference pictures: currently derived as
 - RefPicList1[0] and RefPicList0[0] for B slices and RefPicList0[0] for P slices
 - In side each picture, the blocks are checked in the following order:
 - BR (bottom-right), CT3 (center 3) and CO2 (corner 2)



Proposal

- Order based termination

- The above mentioned neighboring blocks are checked in order.
- Similar to 3D-HEVC, temporal neighboring blocks are checked first and the spatial neighboring blocks are checked afterwards.
- Once a block contains an available disparity motion vector, the derivation process terminates.

Results

- In this section, simulation results are provided with different configurations and compared with MVC+D and 3D-AVC CTC respectively.
 - “BVSP off” indicates that no texture view is coded based on any depth view.
 - “BVSP on” indicates that no texture view is coded based on any depth view of a non-base view, but non-base texture views are coded with access to the depth view components of the base view.

Proposal (BVSP off) vs MVC+D

- The proposed method when BVSP is off, outperforms MVC+D for 22% for texture only coding and 27% for the overall (evaluated by the synthesized views).
- Multi-view compatible

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesised PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	35.1	-1.0	-2.8	0.1	40.3	-1.1	39.4	-1.1
S02	8.4	-0.2	-14.8	0.7	12.7	-0.4	12.0	-0.4
S03	18.0	-0.6	-17.9	1.6	17.2	-0.6	21.6	-0.6
S04	20.4	-0.7	18.8	-1.0	22.3	-0.7	23.6	-0.7
S05	25.5	-1.1	-11.3	0.6	39.6	-1.6	35.7	-1.3
S06	33.5	-1.5	-8.5	0.4	41.8	-1.8	35.4	-1.4
S08	16.0	-0.6	-22.1	1.1	30.3	-1.1	21.6	-0.7
Average	22.4	-0.8	-8.4	0.5	29.2	-1.0	27.0	-0.9

Proposal (BVSP on) vs MVC+D

- After enabling BVSP, the proposed method outperforms MVC+D for 27% for texture only coding and 31% for the overall (evaluated by the synthesized views).

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	41.1	-1.1	-2.5	0.1	46.0	-1.2	44.3	-1.2
S02	12.9	-0.4	-15.2	0.7	17.0	-0.5	15.4	-0.4
S03	22.1	-0.7	-18.2	1.6	20.9	-0.7	25.1	-0.7
S04	30.0	-1.0	20.4	-1.1	31.5	-1.0	31.9	-1.0
S05	28.3	-1.2	-11.5	0.6	41.6	-1.6	38.8	-1.4
S06	35.9	-1.6	-8.6	0.4	43.9	-1.8	37.8	-1.5
S08	16.6	-0.7	-21.8	1.0	30.8	-1.1	22.2	-0.7
Average	26.7	-0.9	-8.2	0.5	33.1	-1.1	30.8	-1.0

Proposal (BVSP on) vs 3D-AVC (depth-first)

- The proposed provides the same performance as 3D-AVC best performing configuration, when evaluated by the overall bitrates and the synthesized views.

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	0.3	0.0	-4.4	0.3	0.0	0.0	-0.5	0.0
S02	0.1	0.0	-6.4	0.3	-0.6	0.0	-0.6	0.0
S03	0.1	0.0	-9.4	0.8	-0.3	0.0	-1.4	0.0
S04	0.2	0.0	-13.1	0.9	-0.7	0.0	-1.4	0.0
S05	1.6	-0.1	-3.4	0.2	0.8	0.0	0.3	0.0
S06	1.1	-0.1	-7.9	0.4	0.2	0.0	0.1	0.0
S08	0.6	0.0	-4.4	0.2	-0.1	0.0	-0.3	0.0
Average	0.6	0.0	-7.0	0.4	-0.1	0.0	-0.6	0.0

Proposal (BVSP on) vs 3D-AVC (depth-first)

- The proposed method when BVSP is off, JCT3V-D0186 outperforms JCT3V-D0185 by 2.2% for texture only coding and 1.8% for the overall (evaluated by the synthesized views).
- Multi-view compatible

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	-5.7	0.2	-0.4	0.0	-5.2	0.2	-4.7	0.2
S02	-0.5	0.0	0.2	0.0	-0.4	0.0	-0.3	0.0
S03	-1.0	0.0	-0.3	0.0	-0.9	0.0	-0.8	0.0
S04	-0.6	0.0	-0.1	0.0	-0.6	0.0	-0.4	0.0
S05	-2.4	0.1	-0.2	0.0	-2.1	0.1	-1.9	0.1
S06	-2.8	0.1	0.1	0.0	-2.5	0.1	-2.1	0.1
S08	-2.5	0.1	-0.1	0.0	-2.2	0.1	-2.1	0.1
Average	-2.2	0.1	-0.1	0.0	-2.0	0.1	-1.8	0.1

Proposal (BVSP on) vs 3D-AVC (depth-first)

- After enabling BVSP, the proposed JCT3V-D0186 outperforms JCT3V-D0185 by 0.1% for texture only coding and 0.1% for the overall (evaluated by the synthesized views).

	Texture Coding		Depth Coding		Total (Coded PSNR)		Total (Synthesed PSNR)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	-0.4	0.0	-0.2	0.0	-0.4	0.0	-0.3	0.0
S02	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0
S03	0.0	0.0	-0.1	0.0	0.0	0.0	0.0	0.0
S04	0.0	0.0	-0.1	0.0	0.0	0.0	0.0	0.0
S05	0.0	0.0	0.0	0.0	0.0	0.0	-0.1	0.0
S06	-0.2	0.0	0.0	0.0	-0.2	0.0	-0.2	0.0
S08	-0.2	0.0	-0.3	0.0	-0.2	0.0	-0.2	0.0
Average	-0.1	0.0	-0.1	0.0	-0.1	0.0	-0.1	0.0

Conclusions

- To avoid relying on depth view component of the same view when decoding each texture view component of a non-base view, NBDV as in 3D-HEVC is used in 3D-AVC with simplifications to derive the disparity vector required for coding tools
 - One single tool set for all configurations
- The proposed method enables all the existing coding tools thus even provides slight gain compared with the best performing 3D-AVC
- The proposed method provides 22% gain for the texture-only coding compared to MVC+D
 - 27% gain for the overall, as evaluated by the total bitrate and the PSNR of the synthesized views.
- When enabling BVSP, 4~6% additional gain can be achieved to reach the best coding performance. There is no need to decode each depth view component for a texture view component in the same non-base view
- With the proposed method, depth-first coding becomes unnecessary.

Special thanks to the following companies
for cross checking these two proposals:

MERL,

MediaTek,

ETRI