



REDEFINING MOBILITY



JCT3V-E0136: CE7: MB-level NBDV for 3D-AVC

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

Abstract

- Neighboring Block based Disparity Vector (NBDV) is proposed for 3D-AVC to significantly improve the coding performance of texture-first coding mode, which meets the MPEG requirement of stereo/multiview compatibility.
- A follow up of JCT3V-D0185 with further simplifications
 - Only one temporal neighboring block is checked in NBDV process
 - Only the horizontal component of disparity motion vector of a neighboring block is checked
 - NBDV process is not applied for intra-coded macroblocks

Abstract

■ Summary of coding performance

			Multiview compatible mode*		Best performing mode**	
	Anchor	Tested	Texture Coding	Total (Synthesed PSNR)	Texture Coding	Total (Synthesed PSNR)
Coding Performance	MVC+D	3D-AVC	-1.15%	-1.55%	-21.52%	-17.80%
	MVC+D	Proposal	-18.31%	-15.18%	-21.00%	-17.49%
	3D-AVC	Proposal	-17.33%	-13.85%	0.67%	0.39%
	3D-AVC	MVC+D	1.17%	1.58%	28.21%	22.17%
	Proposal	MVC+D	23.08%	18.32%	27.34%	21.71%
	Proposal	3D-AVC	21.61%	16.47%	-0.67%	-0.38%

- The proposed method achieves -17.3% coding gain compared to 3D-AVC for multiview compatible mode
- The proposed method achieves comparable coding gain of 3D-AVC for best performing mode

* Both proposal and 3D-AVC using texture first coding, no dependency on depth views

** 3D-AVC using CTC and proposal using texture first coding and dependent on base depth view

Summary

- Summary of complexity measurement

	Proposal compared to 3D-AVC CTC			
	Coding gain	Memory increase	Memory access increase	Ratio of numbers of decoded views to decode all texture views
Proposal in multiview compatible mode	-18.31%*	0.28%	0.1%	Half (3 vs 6)
Proposal in best performing mode	0.39% (for synth. views)	0.28%	0.2%	2/3 (4 vs 6)

*Coding gain of proposal in multiview compatible mode compared to MVC+D for texture views

- Thanks to MERL, MediaTek, ETRI and KHU for the crosscheck! (JCT3V-E0248/E0276/E0289)

Thank you!