

JCT3V-E0128

CE1.h: Adaptive Virtual Depth Block Partition for View Synthesis Prediction and Complexity Analysis

National Cheng Kung University

Chun-Fu Chen, Gwo Giun Lee, Bo-Syun Li

Hong Kong Applied Science and Technology Research Institute

Chunhui Cui, Yan Huo



Abstract

- ▶ In this proposal
 - Adaptively partition the depth block to derive the disparity vectors
 - Using depth distribution to adaptively choose subblock size
 - Supported block size
 - 16x16, 16x8, 8x16, 8x8, 8x4, 4x8
 - Data transfer → up to 28.85% reduction
 - Number of storage accessing → 42.86% ~ 93.63% reduction
 - Slight coding loss
- ▶ Experimental results

	video PSNR / video bitrate	video PSNR / total bitrate	synth PSNR / total bitrate	enc time	dec time
Method 1	0.04%	0.05%	0.03%	99.73%	100.13%
Method 2	0.03%	0.03%	0.02%	99.73%	100.15%

Outline

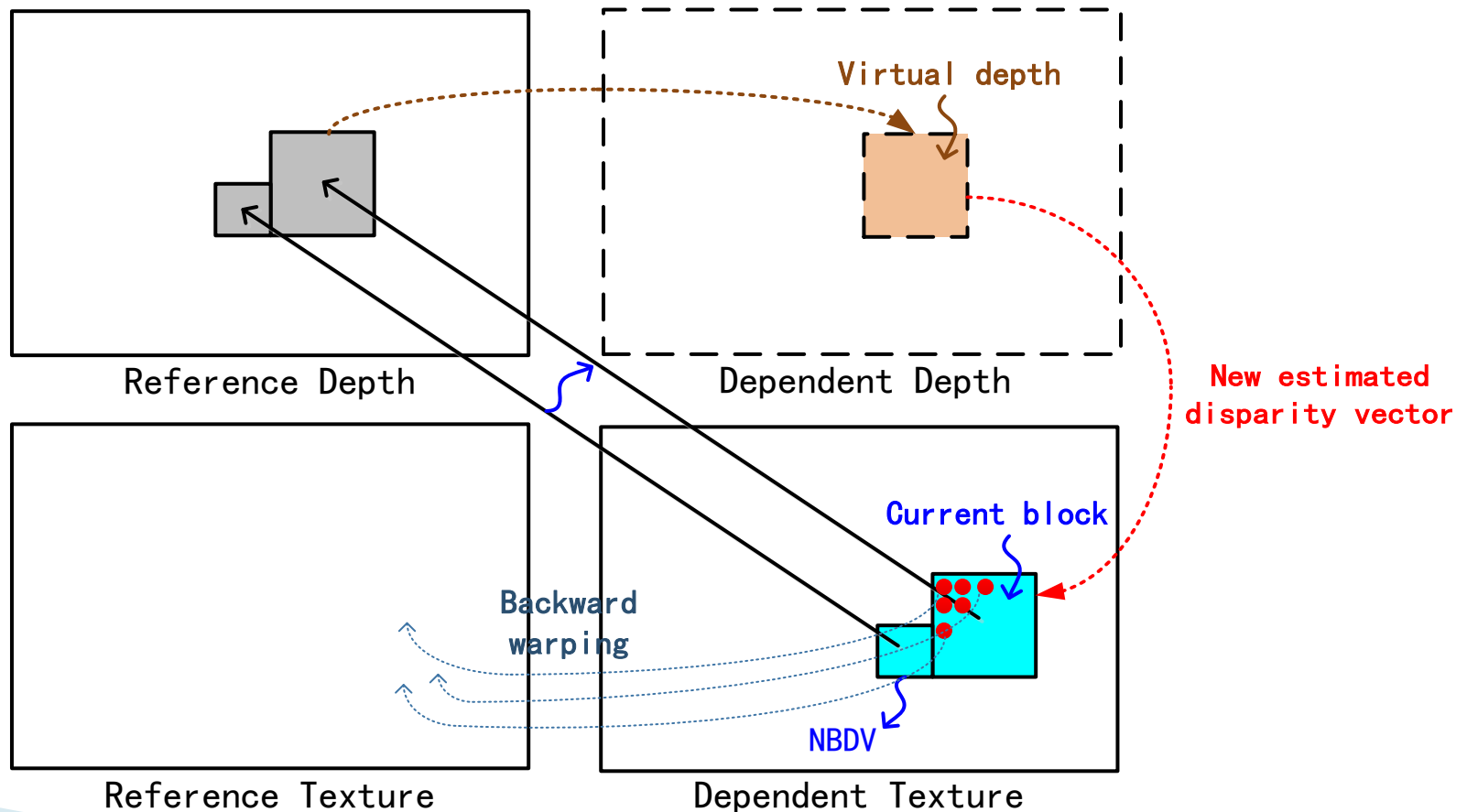
- ▶ Introduction to Backward View Synthesis Prediction
- ▶ Motivation
- ▶ Proposed Algorithm
- ▶ Complexity Analysis
- ▶ Conclusion

Outline

- ▶ Introduction to Backward View Synthesis Prediction
- ▶ Motivation
- ▶ Proposed Algorithm
- ▶ Complexity Analysis
- ▶ Conclusion

Introduction

► Backward View Synthesis Prediction (B-VSP)



Outline

- ▶ Introduction to Backward View Synthesis Prediction
- ▶ **Motivation**
- ▶ Proposed Algorithm
- ▶ Complexity Analysis
- ▶ Conclusion

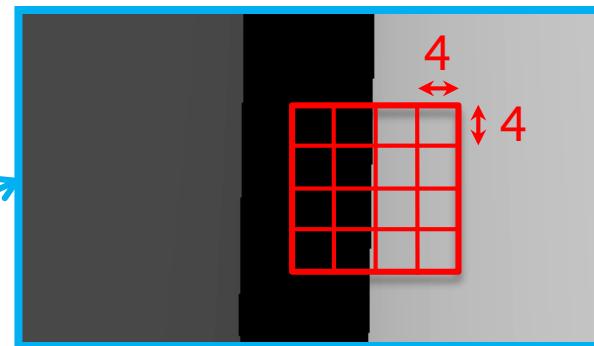
Motivation

- ▶ Adaptively partition the virtual depth block
 - Accurate disparity vectors at appropriate block size
 - Data transfer rate reduction
 - Number of storage accessing reduction



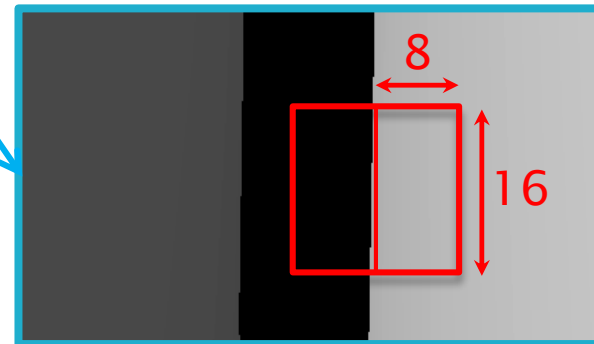
Original depth
(frame #247, view 05)

DV: Disparity vector



Block partitioned (Anchor)

16 DVs



Block partitioned (Proposal)

2 DVs

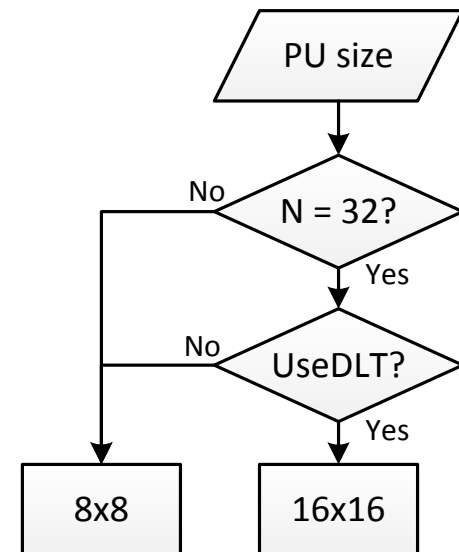
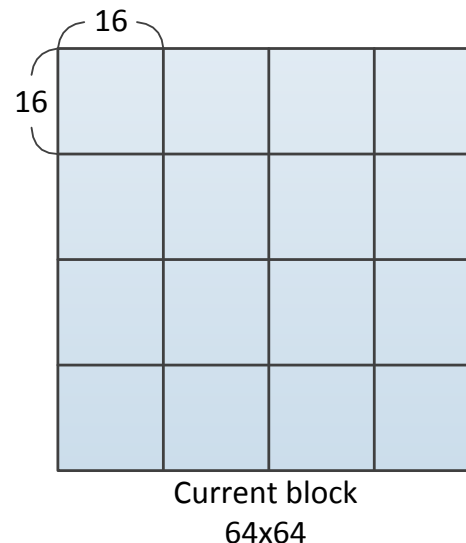
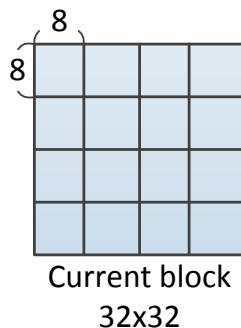
Outline

- ▶ Introduction to Backward View Synthesis Prediction
- ▶ Motivation
- ▶ **Proposed Algorithm**
- ▶ Complexity Analysis
- ▶ Conclusion

Proposed Algorithm (2/4)

► Step 1: Subblock partition

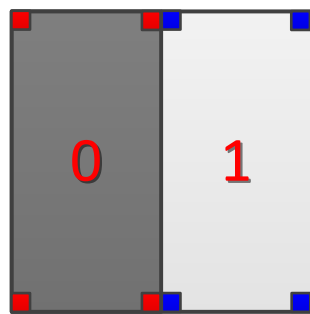
- According to the depth distribution (UseDLT flag)
- Prediction unit (PU) size
 - $N = 32 \rightarrow 16 \times 16$ or 8×8
 - $N = 4, 8, 16 \rightarrow 8 \times 8$



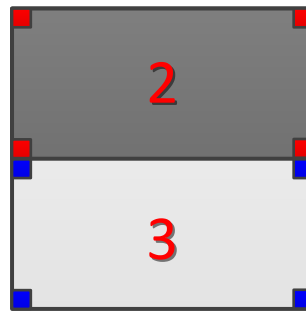
PU size: $2N \times 2N$, $2N \times N$, $N \times 2N$
DLT: Depth Lookup Table

Proposed Algorithm (3/4)

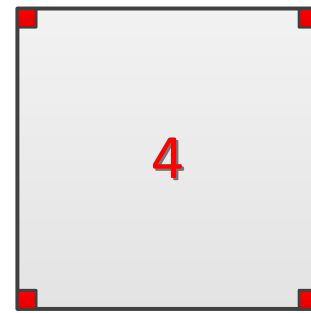
- ▶ **Step 2:** Using supported partition types to generate representative depth candidates, five candidates
 - Supported subblock partition type
 - Subblock $16 \times 16 \rightarrow 16 \times 16, 16 \times 8, 8 \times 16$
 - Subblock $8 \times 8 \rightarrow 8 \times 8, 8 \times 4, 4 \times 8$
 - Maximum depth out of four corners



$M/2 \times M$



$M \times M/2$



$M \times M$

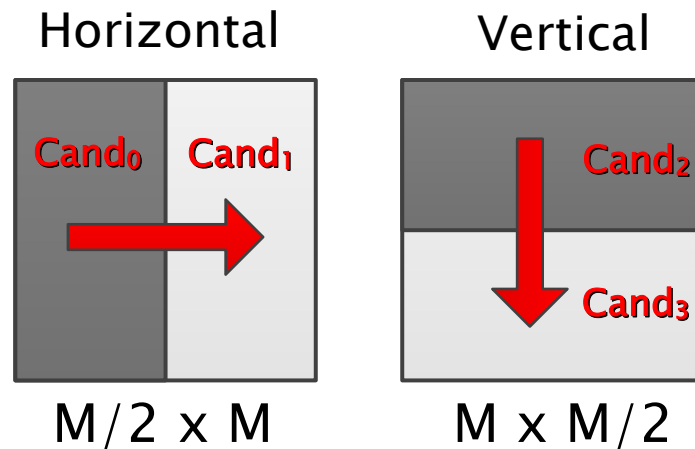
$M \rightarrow 8 \text{ or } 16$

Generating five candidates

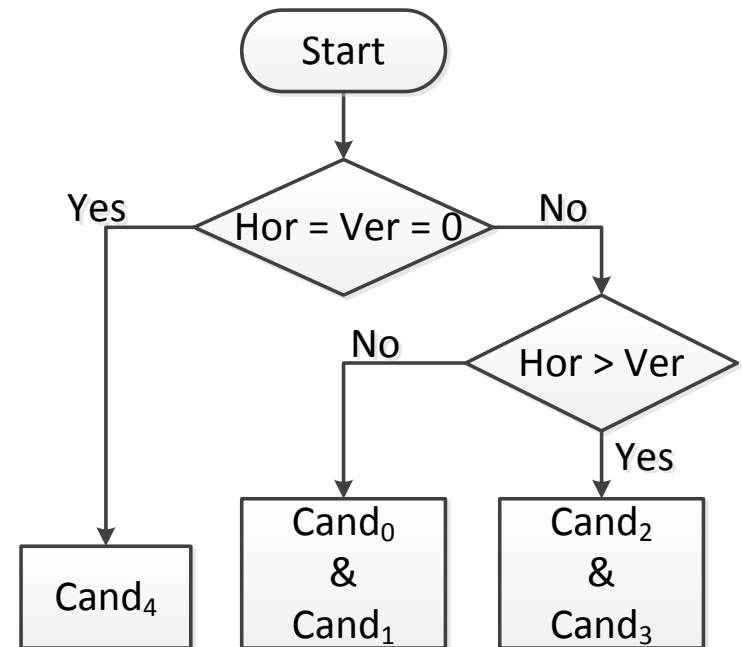
Proposed Algorithm (3/4)

- ▶ **Step 3:** Horizontal and vertical change detection
 - Reuse possible representative depth (five candidates) to determine appropriate partition type

$$\text{Hor} = |\text{Cand}_0 - \text{Cand}_1|$$
$$\text{Ver} = |\text{Cand}_2 - \text{Cand}_3|$$



$M \rightarrow 8 \text{ or } 16$



Experimental Results (1 / 2)

- ▶ Method 1: According to depth distribution (UseDLT) to choose 8x8 of 16x16 subblock size

	video 0	video 1	video 2	video PSNR / video bitrate	video PSNR / total bitrate	synth PSNR / total bitrate	enc time	dec time	ren time
Balloons	0.00%	0.06%	-0.03%	0.01%	0.01%	0.00%	99.01%	100.10%	98.38%
Kendo	0.00%	0.03%	-0.11%	0.00%	0.02%	0.00%	98.88%	100.01%	97.46%
Newspaper_CC	0.00%	0.00%	-0.14%	-0.02%	0.00%	0.02%	99.07%	99.92%	98.00%
GT_Fly	0.00%	0.50%	0.32%	0.09%	0.09%	0.07%	100.05%	100.62%	97.43%
Poznan_Hall2	0.00%	-0.11%	0.20%	0.01%	0.03%	0.07%	100.49%	100.04%	100.14%
Poznan_Street	0.00%	0.04%	0.01%	0.01%	0.01%	0.01%	100.21%	99.99%	97.34%
Undo_Dancer	0.00%	0.77%	0.61%	0.19%	0.17%	0.02%	100.36%	100.20%	100.08%
1024x768	0.00%	0.03%	-0.09%	0.00%	0.01%	0.00%	98.99%	100.01%	97.95%
1920x1088	0.00%	0.30%	0.28%	0.07%	0.08%	0.04%	100.28%	100.21%	98.75%
average	0.00%	0.18%	0.12%	0.04%	0.05%	0.03%	99.73%	100.13%	98.41%

Method 1 vs HTM-7.0r1

Experimental Results (2/2)

- ▶ Method 2: Only support 8x8 subblock size (Without UseDLT flag)

	video 0	video 1	video 2	video PSNR / video bitrate	video PSNR / total bitrate	synth PSNR / total bitrate	enc time	dec time	ren time
Balloons	0.00%	0.03%	-0.15%	-0.02%	-0.01%	-0.05%	100.46%	100.13%	99.52%
Kendo	0.00%	0.18%	-0.06%	0.02%	0.01%	0.03%	100.15%	99.94%	98.38%
Newspaper_CC	0.00%	-0.04%	-0.11%	-0.02%	-0.01%	0.06%	100.33%	99.97%	98.72%
GT_Fly	0.00%	0.50%	0.32%	0.09%	0.09%	0.07%	98.03%	100.81%	96.91%
Poznan_Hall2	0.00%	-0.18%	-0.13%	-0.07%	-0.06%	-0.01%	100.38%	100.03%	100.08%
Poznan_Street	0.00%	0.04%	0.01%	0.01%	0.01%	0.01%	100.51%	100.01%	99.49%
Undo_Dancer	0.00%	0.77%	0.61%	0.19%	0.17%	0.02%	98.24%	100.13%	95.37%
1024x768	0.00%	0.06%	-0.11%	-0.01%	0.00%	0.01%	100.31%	100.01%	98.87%
1920x1088	0.00%	0.28%	0.20%	0.05%	0.05%	0.02%	99.29%	100.24%	97.96%
average	0.00%	0.19%	0.07%	0.03%	0.03%	0.02%	99.73%	100.15%	98.35%

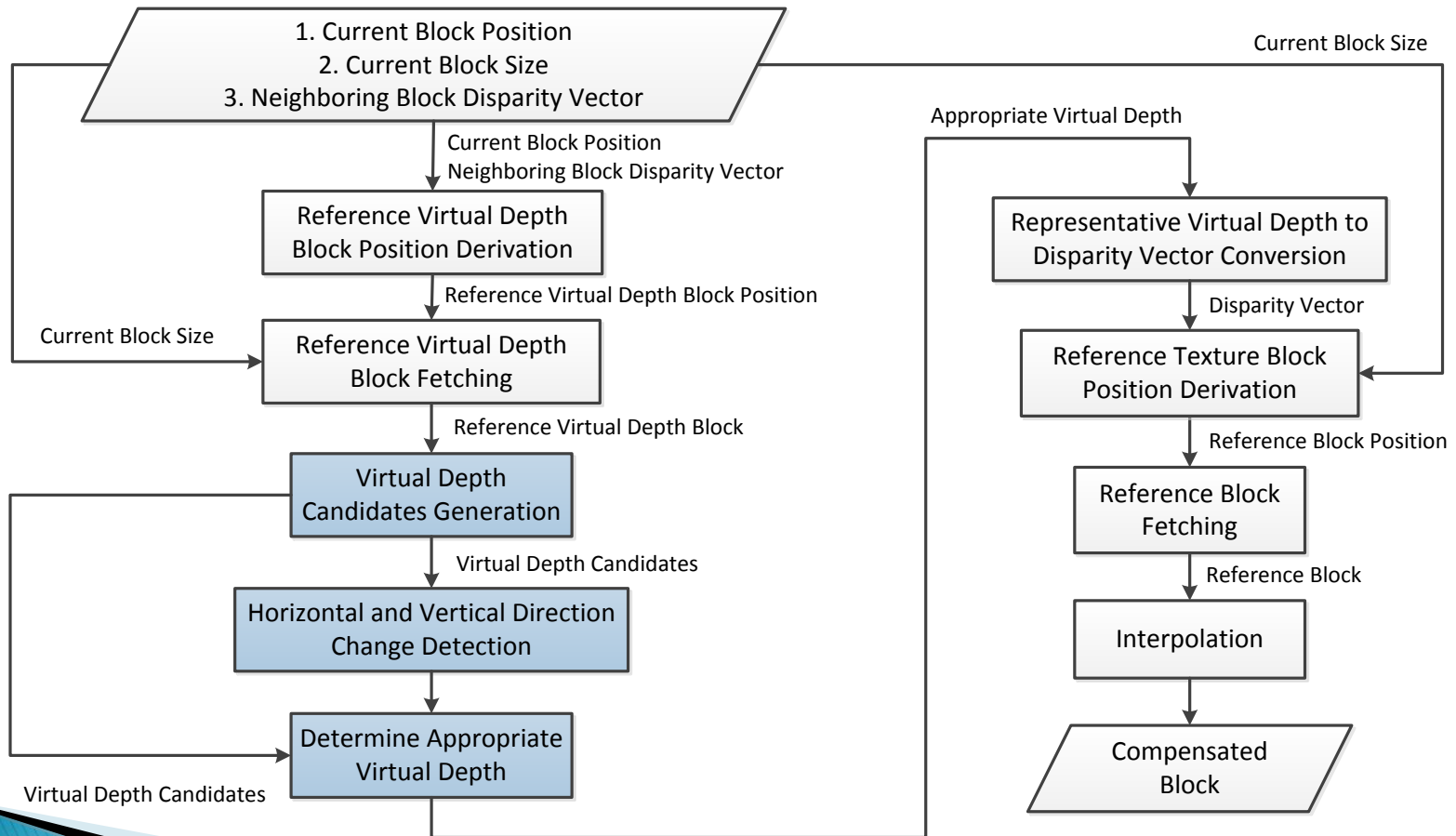
Method 2 vs HTM-7.0r1

Outline

- ▶ Introduction to Backward View Synthesis Prediction
- ▶ Motivation
- ▶ Proposed Algorithm
- ▶ **Complexity Analysis**
- ▶ Conclusion

Complexity Analysis (1 / 9)

► Flowchart of proposed B-VSP



Complexity Analysis (2 / 9)

- ▶ Number of operations (data granularity is one PU)

- Virtual depth candidates generation

$$Cand_n = \underset{\substack{i=0, \text{SupportPartitionSize}_H-1 \\ j=0, \text{SupportPartitionSize}_V-1}}{MAX} (Depth[i, j]), \quad n = 0, 1, \dots, 4$$

- Horizontal and vertical change detection and determine appropriate Virtual depth

$$M_1 = |Cand_0 - Cand_1|, \quad M_2 = |Cand_2 - Cand_3|$$

If $M_1 = M_2$, then choose $Cand_4$ (16×16)

else is $M_1 > M_2$, then $Cand_2$ and $Cand_3$ (16×8)

else $Cand_0$ and $Cand_1$ (8×16)

- Interpolation

Luma:

2/4 pixel position: filter = [-1, 4, -11, 40, 40, 5, -11, 5, -1]

Chroma:

1/8 pixel position: filter = [-2, 58, 10, -2]

Complexity Analysis (3 / 9)

► Data storage requirement for one PU (bits)

	Proposed B-VSP	Anchor
Reference virtual depth block	$W \times H \times 8$	$W \times H \times 8$
Virtual depth candidates	4×8	N/A
Disparity Vector	2×12	1×12
Reference texture block with extended taps	$(3 + M + 4) \times M \times 8$ or $(3 + (M/2) + 4) \times M \times 8$ or $(3 + M + 4) \times (M/2) \times 8$	$(3 + 4 + 4) \times 4 \times 8$
Compensated block	$W \times H \times 8$	$W \times H \times 8$

W is the block width and H is the block height for PU size;
 M is the block width and height for the subblock size;
 The bits of one pixel of texture or depth is 8 bits and disparity is 12 bits

Complexity Analysis (4 / 9)

► Data transfer for one PU (bits/PU)

	Proposed B-VSP	Anchor
Reference virtual depth block	$W \times H \times 8$	$W \times H \times 8$
Reference texture block with extended taps	$(3+M+4) \times M \times (W/8) \times (H/8) \times 8 + 2 \times$ $[1+(M/2)+2] \times (M/2) \times (W/4) \times (H/4) \times 8$ <p>or</p> $(3+M+4) \times M \times (W/16) \times (H/16) \times 8 + 2 \times$ $[1+(M/2)+2] \times (M/2) \times (W/8) \times (H/8) \times 8$	$(3+4+4) \times 4 \times (W/4) \times (H/4) \times 8 + 2 \times$ $[1+2+2] \times 2 \times (W/2) \times (H/2) \times 8$
Compensated block	$W \times H \times 8 + 2 \times (W/2) \times (H/2) \times 8$	$W \times H \times 8 + 2 \times (W/2) \times (H/2) \times 8$

W is the block width and H is the block height for PU size;

M is the block width and height for the subblock size;

The bits of one pixel of texture or depth is 8 bits and disparity is 12 bits

Complexity Analysis (5 / 9)

► Number of operations

PU size	Proposal (Method 1)						Anchor	Ratio (to anchor)
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	
64x64	35169	35169	35169	36225	36225	36225	36353	
64x32	35170	35170	35170	36226	36226	36226	36354	
32x64	35170	35170	35170	36226	36226	36226	36354	
32x32	N/A	N/A	N/A	36228	36228	36228	36356	
32x16	N/A	N/A	N/A	36232	36232	36232	36360	
16x32	N/A	N/A	N/A	36232	36232	36232	36360	
16x16	N/A	N/A	N/A	36240	36240	36240	36368	
16x8	N/A	N/A	N/A	36256	36256	36256	36384	
8x16	N/A	N/A	N/A	36256	36256	36256	36384	
8x8	N/A	N/A	N/A	36288	36288	36288	36416	
8x4	N/A	N/A	N/A	N/A	35712	N/A	36480	
4x8	N/A	N/A	N/A	N/A	N/A	35712	36480	
							Worst	99.47%

Add/Sub/Comp

Complexity Analysis (6 / 9)

► Data storage requirement (Bytes)

PU size	Proposal (Method 1)						Anchor	Ratio (to anchor)
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	
64x64	8565.5	8383	8439	8317.5	8259	8287	8237.5	
64x32	4469.5	4287	4343	4221.5	4163	4191	4141.5	
32x64	4469.5	4287	4343	4221.5	4163	4191	4141.5	
32x32	N/A	N/A	N/A	2173.5	2115	2143	2093.5	
32x16	N/A	N/A	N/A	1149.5	1091	1119	1069.5	
16x32	N/A	N/A	N/A	1149.5	1091	1119	1069.5	
16x16	N/A	N/A	N/A	637.5	579	607	557.5	
16x8	N/A	N/A	N/A	381.5	323	351	301.5	
8x16	N/A	N/A	N/A	381.5	323	351	301.5	
8x8	N/A	N/A	N/A	253.5	195	223	173.5	
8x4	N/A	N/A	N/A	N/A	126.5	N/A	109.5	
4x8	N/A	N/A	N/A	N/A	N/A	154.5	109.5	
							Worst	103.98%

Data granularity One LCU

Complexity Analysis (7 / 9)

► Data transfer (Bytes)

PU size	Proposal (Method 1)						Anchor	Ratio (to anchor)
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	
64x64	18944	18944	21504	21504	21504	26624	26624	
64x32	18944	18944	21504	21504	21504	26624	26624	
32x64	18944	18944	21504	21504	21504	26624	26624	
32x32	N/A	N/A	N/A	21504	21504	26624	26624	
32x16	N/A	N/A	N/A	21504	21504	26624	26624	
16x32	N/A	N/A	N/A	21504	21504	26624	26624	
16x16	N/A	N/A	N/A	21504	21504	26624	26624	
16x8	N/A	N/A	N/A	21504	21504	26624	26624	
8x16	N/A	N/A	N/A	21504	21504	26624	26624	
8x8	N/A	N/A	N/A	21504	21504	26624	26624	
8x4	N/A	N/A	N/A	N/A	21504	N/A	26624	
4x8	N/A	N/A	N/A	N/A	N/A	26624	26624	
							Worst	100.00%
							Average	84.52%
							Best	71.15%

Complexity Analysis (8/9)

► Number of storage accessing

PU size	Proposal (Method 1)						Anchor	Ratio (to anchor)
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	
64x64	49	97	97	193	385	385	769	
64x32	50	98	98	194	386	386	770	
32x64	50	98	98	194	386	386	770	
32x32	N/A	N/A	N/A	196	388	388	772	
32x16	N/A	N/A	N/A	200	392	392	776	
16x32	N/A	N/A	N/A	200	392	392	776	
16x16	N/A	N/A	N/A	208	400	400	784	
16x8	N/A	N/A	N/A	224	416	416	800	
8x16	N/A	N/A	N/A	224	416	416	800	
8x8	N/A	N/A	N/A	256	448	448	832	
8x4	N/A	N/A	N/A	N/A	512	N/A	896	
4x8	N/A	N/A	N/A	N/A	N/A	512	896	
							Worst	57.14%
							Average	35.99%
							Best	6.37%

Complexity Analysis (9/9)

► Summary report

Ratio to anchor

	# of operations Add/Sub/Comp	Data storage	Data transfer	# of storage accessing
Worst	99.47%	103.98%	100.00%	57.14%
Average	–	–	84.52%	35.99%
Best	–	–	71.15%	6.37%

Method 1

Outline

- ▶ Introduction to Backward View Synthesis Prediction
- ▶ Motivation
- ▶ Proposed Algorithm
- ▶ Complexity Analysis
- ▶ Conclusion

Conclusion

- ▶ According to coding performance and complexity analysis, it is recommended to adopt proposed method 1 to 3D-HEVC.

Ratio to anchor

	Coding performance			Complexity analysis				
	video PSNR vs. video bitrate	video PSNR vs. total bitrate	synth PSNR vs. total bitrate	Scenario	# of operations	Data storage	Data transfer	# of storage accessing
Method 1	0.04%	0.05%	0.03%	Worst	99.47%	103.98%	100.00%	57.14%
				Average	–	–	84.52%	35.99%
				Best	–	–	71.15%	6.37%
Method 2	0.03%	0.03%	0.02%	Worst	99.47%	100.97%	100.00%	57.14%
				Average	–	–	87.38%	43.21%
				Best	–	–	80.77%	25.10%

Thanks for your attention!

