

# JCT3V-D0089: 3D-HEVC: Adaptive Virtual Depth Block Partition for View Synthesis Prediction and Corresponding Complexity Analysis

National Cheng Kung University (NCKU)  
Chun-Fu Chen, Gwo Giun (Chris) Lee, Bo-Syun Li

Hong Kong Applied Science and Technology Research  
Institute (ASTRI)  
Chunhui Cui, Yan Huo

# Abstract

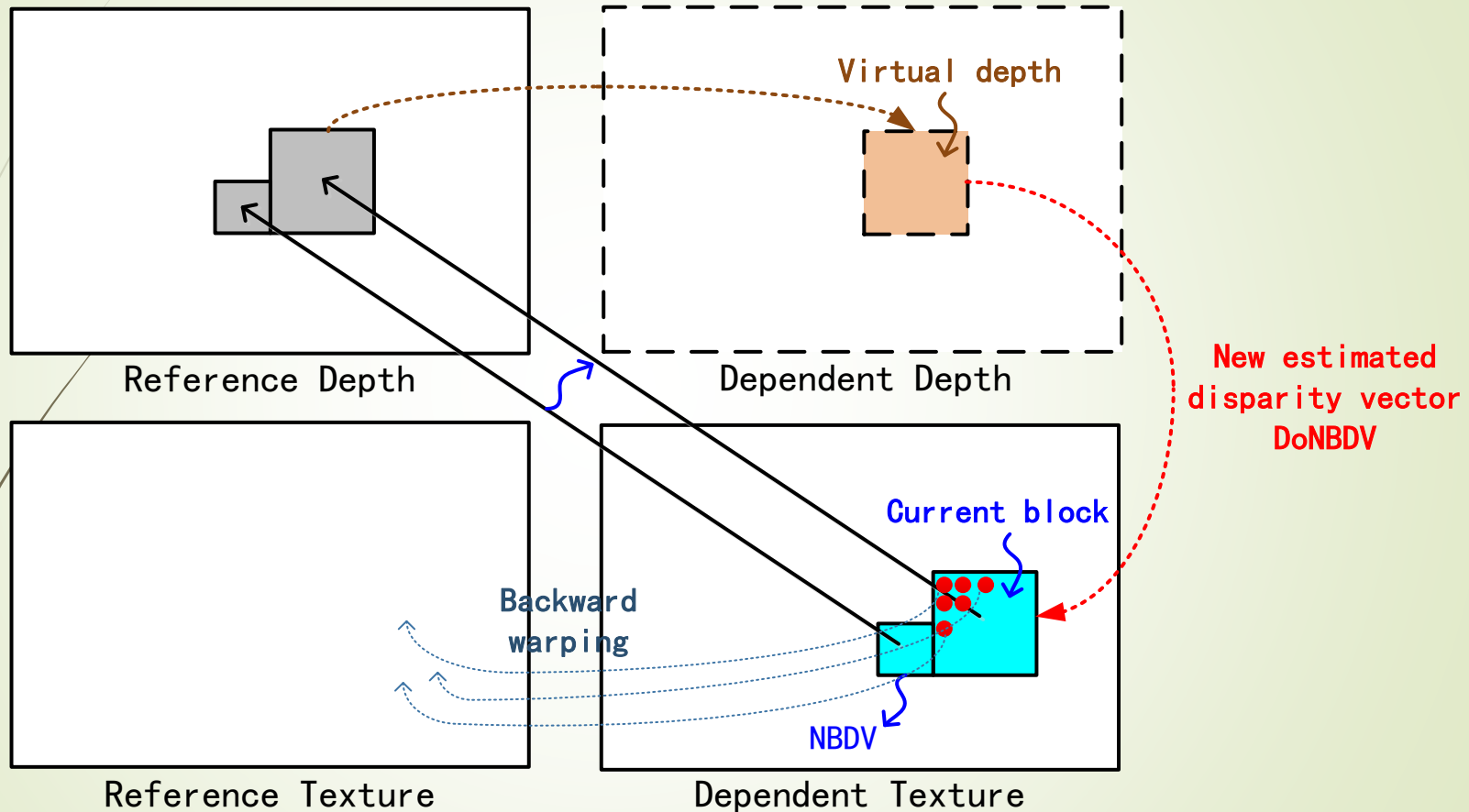
- In this proposal
  - Adaptively partition the depth block to derive the disparity vectors (DoNBDV)
  - Supported block size: 8x8, 8x4, 4x8 block
  - The complexity analysis of test 04 is provided.
- Experimental results

	video PSNR / video bitrate	video PSNR / total bitrate	synth PSNR / total bitrate	dec time
Test 01	0.0%	0.0%	0.0%	101.3%
Test 02	0.0%	0.0%	0.0%	100.2%
Test 03	0.0%	0.0%	0.0%	100.4%
Test 04	0.1%	0.0%	0.0%	100.2%

# Outline

- Abstract
- Introduction to Backward View Synthesis Prediction
- Motivation
- Proposed Algorithm
- Experimental Results
- Complexity Assessment
- Conclusion

# Introduction to Backward View Synthesis Prediction (B-VSP)

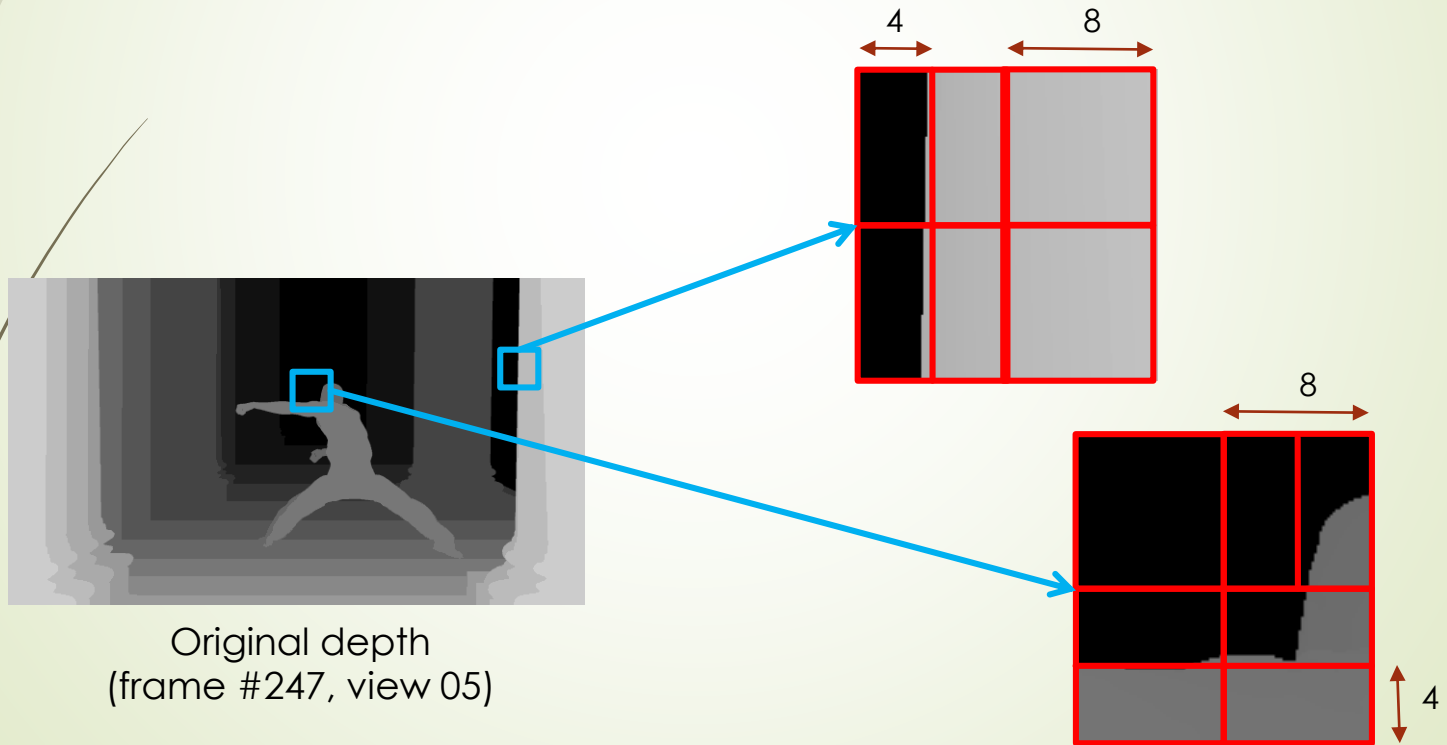


NBDV: Neighboring block based disparity vector

DoNBDV: Depth-oriented neighboring block based disparity vector

# Motivation

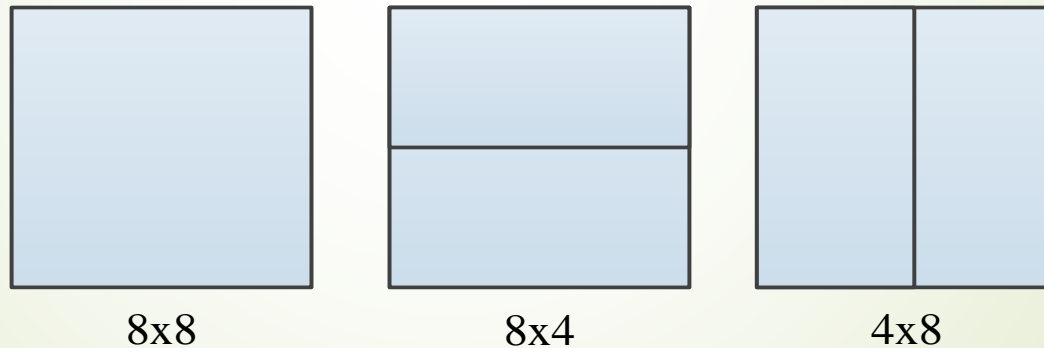
- ▶ Block size becomes larger  $\rightarrow$  DV may be inaccurate
- ▶ Adaptive block partition to obtain the DVs.



# Proposed Algorithm (1/5)

- ▶ Object: Modify the second disparity vector derivation (DoNBDV) in BVSP
  - ▶ Content adaptively partition depth block to derive the second disparity vector at 8x8 block basis

Three depth block partition types



# Proposed Algorithm (2/5)

- Content adaptive depth block partition
  - According to the gradient of a 8x8 depth block
  - Horizontal depth transition detection

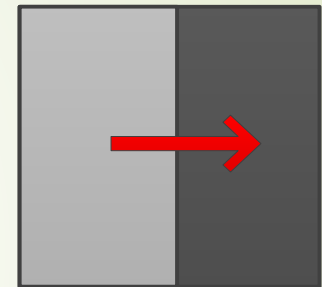
$$\text{NoLC}_H = \sum_{k=0}^N \text{LineChange}[k]$$

$$\text{LineChange}[k] = \begin{cases} 1, & \text{Grad}_H[k] > TH_H \\ 0, & \text{Otherwise} \end{cases}$$

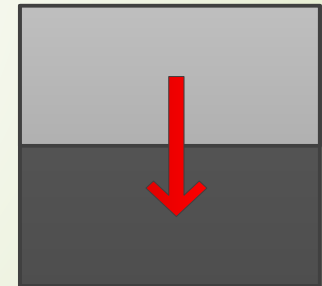
- Vertical depth transition detection

$$\text{NoLC}_V = \sum_{k=0}^N \text{LineChange}[k]$$

$$\text{LineChange}[k] = \begin{cases} 1, & \text{Grad}_V[k] > TH_V \\ 0, & \text{Otherwise} \end{cases}$$



4x8



8x4

$\text{Grad}_H$ : Horizontal gradient

$\text{Grad}_V$ : Vertical gradient

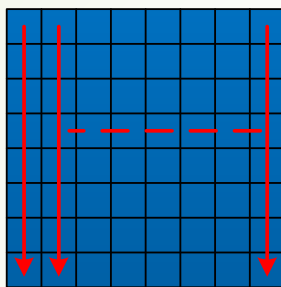
$\text{NoDC}_H$ : Number of depth changes in horizontal

$\text{NoDC}_V$ : Number of depth changes in vertical

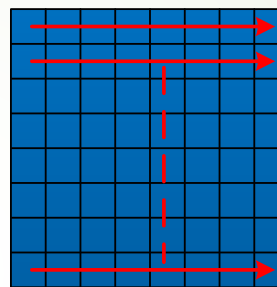
# Proposed Algorithm (3/5)

## Depth block analysis

Method 1: Full-pixels gradient computation

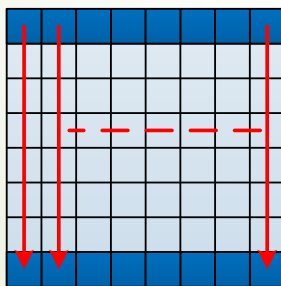


Vertical Direction

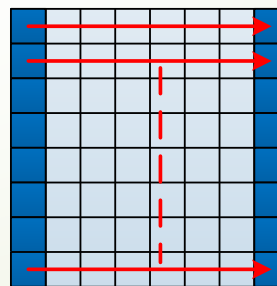


Horizontal Direction

Method 2: Outmost pixels gradient computation



Vertical Direction



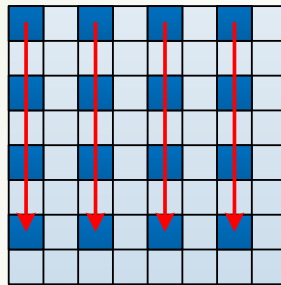
Horizontal Direction



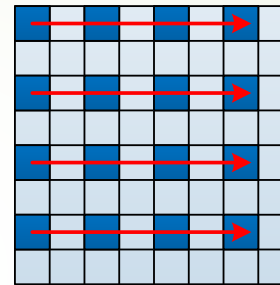
# Proposed Algorithm (4/5)

## Depth block analysis

Method 3: Full- pixels gradient computation with down-sampling by 2

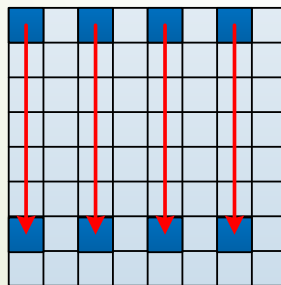


Vertical Direction

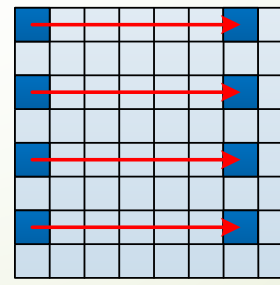


Horizontal Direction

Method 4: Outmost pixels gradient computation with down-sampling by 2



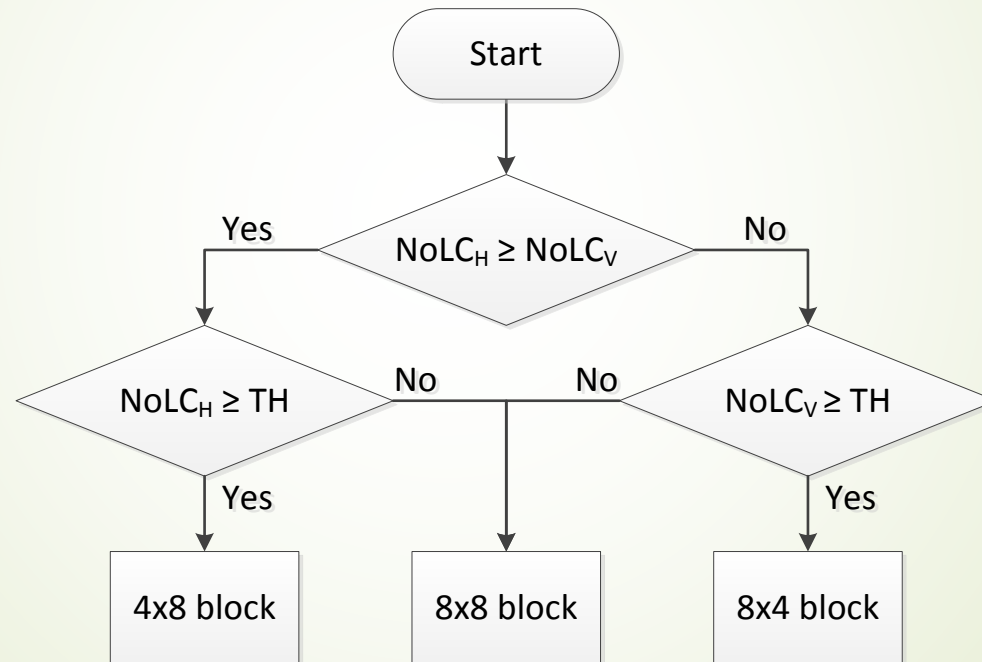
Vertical Direction



Horizontal Direction

# Proposed Algorithm (5/5)

- Content adaptive depth block partition (Cont.)
  - According to the number of line changes in horizontal and vertical to determine partition type



$NoLC_H$ : Number of line changes in horizontal direction  
 $NoLC_V$ : Number of line changes in vertical direction

# Experimental Results (1/4)

11

► Method 01: Full pixels gradient computation

	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	0.0%	-0.1%	0.0%	0.0%	0.0%	-0.1%	100.7%	100.1%	100.0%
Kendo	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%	100.6%	100.3%	99.9%
Newspaper_CC	0.0%	-0.2%	0.1%	0.0%	0.0%	0.0%	100.7%	99.8%	100.0%
GT_Fly	0.0%	0.5%	0.1%	0.1%	0.1%	0.1%	100.6%	105.3%	99.9%
Poznan_Hall2	0.0%	-0.5%	-0.3%	-0.2%	-0.2%	-0.1%	100.6%	100.8%	99.8%
Poznan_Street	0.0%	0.3%	0.2%	0.1%	0.1%	0.1%	100.2%	100.3%	101.7%
Undo_Dancer	0.0%	0.6%	0.7%	0.2%	0.1%	-0.2%	100.5%	102.4%	100.2%
1024x768	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	100.6%	100.1%	100.0%
1920x1088	0.0%	0.2%	0.2%	0.0%	0.0%	-0.1%	100.5%	102.2%	100.4%
<b>average</b>	<b>0.0%</b>	<b>0.1%</b>	<b>0.1%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>100.5%</b>	<b>101.3%</b>	<b>100.2%</b>

# Experimental Results (2/4)

- Method 02: Outmost pixels gradient computation

	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	0.0%	0.0%	0.0%	0.0%	0.0%	-0.1%	100.2%	100.2%	101.1%
Kendo	0.0%	0.2%	0.1%	0.1%	0.0%	0.1%	100.5%	99.8%	99.8%
Newspaper_CC	0.0%	-0.1%	0.1%	0.0%	0.0%	0.0%	100.3%	99.2%	100.3%
GT_Fly	0.0%	0.4%	0.2%	0.1%	0.1%	0.0%	100.3%	102.0%	100.6%
Poznan_Hall2	0.0%	-0.5%	-0.1%	-0.1%	-0.1%	-0.2%	100.1%	99.9%	99.8%
Poznan_Street	0.0%	0.4%	0.3%	0.1%	0.1%	0.1%	100.3%	99.6%	100.7%
Undo_Dancer	0.0%	0.7%	0.8%	0.2%	0.2%	-0.1%	100.1%	100.8%	100.3%
1024x768	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%	100.3%	99.8%	100.4%
1920x1088	0.0%	0.3%	0.3%	0.1%	0.0%	0.0%	100.2%	100.6%	100.3%
<b>average</b>	<b>0.0%</b>	<b>0.2%</b>	<b>0.2%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>100.3%</b>	<b>100.2%</b>	<b>100.4%</b>

# Experimental Results (3/4)

13

- Method 03: Full pixels gradient computation downsampled by 2

	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	0.0%	-0.1%	0.0%	0.0%	0.0%	-0.1%	100.2%	99.9%	99.9%
Kendo	0.0%	0.0%	-0.1%	0.0%	0.0%	0.0%	100.3%	100.1%	99.9%
Newspaper_CC	0.0%	-0.2%	0.0%	0.0%	0.0%	-0.1%	100.3%	99.6%	100.1%
GT_Fly	0.0%	0.4%	0.5%	0.1%	0.1%	0.1%	100.2%	102.4%	99.9%
Poznan_Hall2	0.0%	-0.3%	0.0%	-0.1%	-0.1%	-0.1%	100.0%	100.1%	99.6%
Poznan_Street	0.0%	0.3%	0.4%	0.1%	0.1%	0.1%	99.2%	99.8%	100.5%
Undo_Dancer	0.0%	0.9%	0.9%	0.2%	0.2%	0.0%	100.1%	101.0%	100.4%
1024x768	0.0%	-0.1%	0.0%	0.0%	0.0%	-0.1%	100.3%	99.9%	99.9%
1920x1088	0.0%	0.3%	0.5%	0.1%	0.1%	0.0%	99.9%	100.9%	100.1%
<b>average</b>	<b>0.0%</b>	<b>0.2%</b>	<b>0.2%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>100.1%</b>	<b>100.4%</b>	<b>100.0%</b>

# Experimental Results (4/4)

14

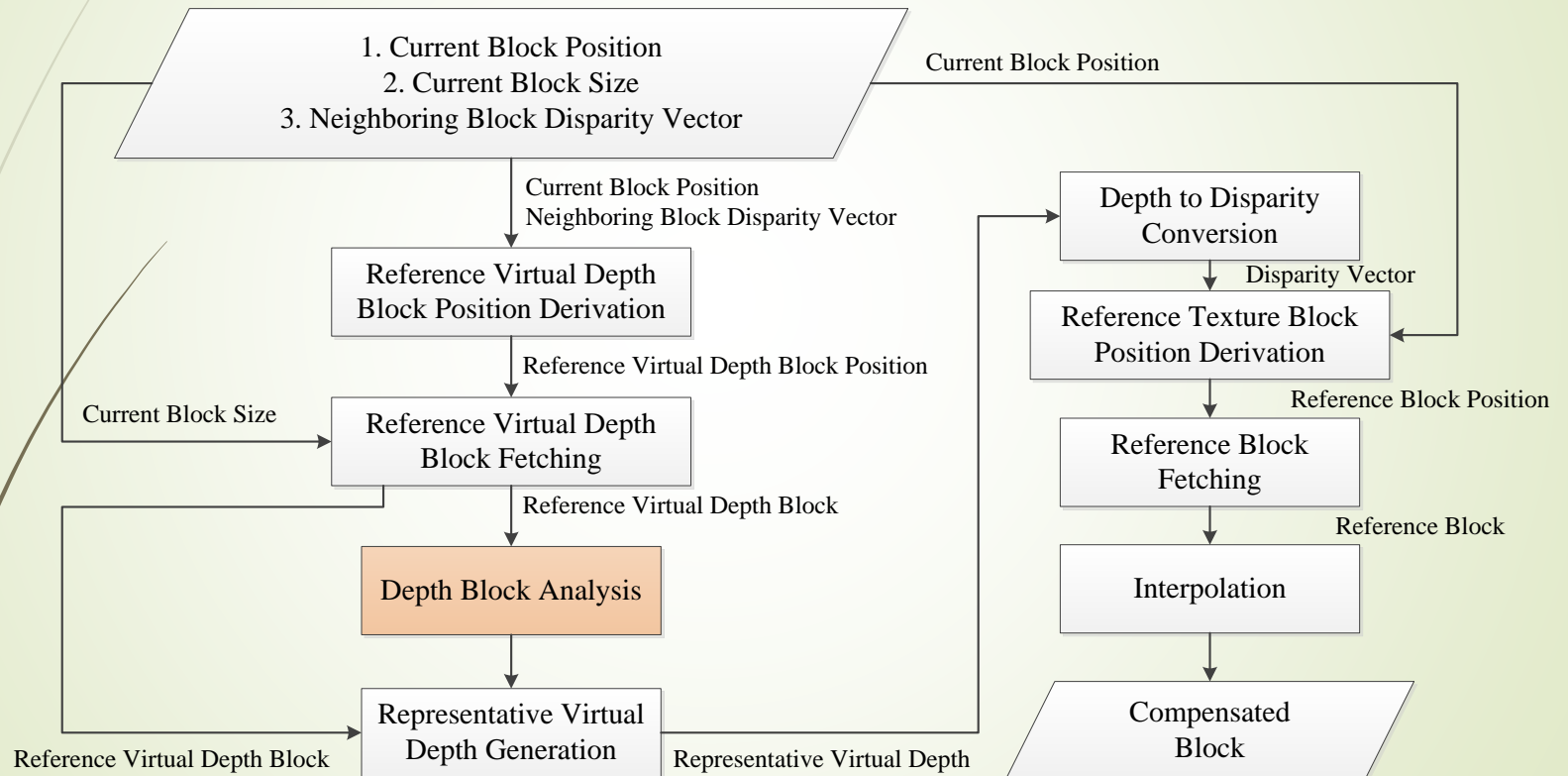
- Method 04: Outmost pixels gradient computation downsampled by 2

	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	0.0%	-0.1%	-0.1%	-0.1%	-0.1%	0.0%	100.1%	99.9%	100.3%
Kendo	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.3%	100.1%	99.5%
Newspaper_CC	0.0%	-0.1%	0.1%	0.0%	0.0%	0.0%	100.1%	99.7%	99.7%
GT_Fly	0.0%	0.7%	0.3%	0.1%	0.1%	0.1%	100.2%	101.6%	100.1%
Poznan_Hall2	0.0%	0.0%	0.2%	0.0%	0.0%	0.1%	100.1%	100.2%	100.1%
Poznan_Street	0.0%	0.5%	0.2%	0.1%	0.1%	0.1%	100.2%	99.4%	100.6%
Undo_Dancer	0.0%	0.7%	1.0%	0.2%	0.2%	0.0%	99.9%	100.5%	99.7%
1024x768	0.0%	-0.1%	0.0%	0.0%	0.0%	0.0%	100.2%	99.9%	99.8%
1920x1088	0.0%	0.5%	0.4%	0.1%	0.1%	0.1%	100.1%	100.4%	100.1%
<b>average</b>	<b>0.0%</b>	<b>0.3%</b>	<b>0.2%</b>	<b>0.1%</b>	<b>0.0%</b>	<b>0.0%</b>	<b>100.1%</b>	<b>100.2%</b>	<b>100.0%</b>



# Complexity Assessment(1/8)

## Flowchart of proposed B-VSP



# Complexity Assessment(2/8)

- Data granularity
  - One PU, including 64x64, 64x32, etc
- Major process of the proposed B-VSP
  - Depth content analysis
  - Representative depth generation
  - Interpolation



# Complexity Assessment(3/8)

- Number of operations (data granularity is one PU)

- Depth content analysis

$$Grad_v[i] = abs(Depth[j, i] - Depth[j + 7, i]), 0 \leq i \leq 3$$

$$Th_v = \left( MAX_{0 \leq i \leq 3} (Grad_v[i]) \right) \gg 1$$

$$NoLC = \sum_{i=0}^3 LineChange[i], \quad LineChange[i] = \begin{cases} 1, & \text{one of } Grad_v[i] > TH_v \\ 0, & \text{otherwise} \end{cases}$$

- Representative virtual depth generation

$$RepresentativeDepth = MAX_{0 \leq j \leq subblock\_H, 0 \leq i \leq subblock\_V} (Depth[j, i])$$

- Interpolation:

$$1/4 \text{ pixel position: } filter_0 = [-1, 4, 10, 58, 17, -5, 1]$$

$$2/4 \text{ pixel position: } filter_1 = [-1, 4, -11, 40, 40, 5, -11, 5, -1]$$

$$3/4 \text{ pixel position: } filter_2 = [1, -5, 17, 58, 10, 4, -1]$$

# Complexity Assessment(4/8)

Data storage requirement for one PU (bits)		
	Proposed B-VSP	Anchor
Reference virtual depth block	$M \times N \times 8$	$M \times N \times 8$
Reference texture block with extended taps	$(8+8) \times 8 \times 8$	$(8+4) \times 4 \times 8$
Compensated block	$M \times N \times 8$	$M \times N \times 8$
Analyzed gradient map and buffer of depth maximums in one $4 \times 4$ block	$((4+4)+4) \times 8$	N/A

M is the block width and N is the block height in horizontal and vertical direction respectively; the bits of one pixel of texture or depth is 8 bits.

# Complexity Assessment(5/8)

19

Data transfer rate for one PU (bits/PU)		
	Proposed B-VSP	Anchor
Reference virtual depth block	$M \times N \times 8$	$M \times N \times 8$
Reference texture block with extended taps	$(8+8) \times 8 \times (M/8) \times (N/8) \times 8$	$(8+4) \times 4 \times (M/4) \times (N/4) \times 8$
Compensated block	$M \times N \times 8$	$M \times N \times 8$

M is the block width and N is the block height in horizontal and vertical direction respectively; the bits of one pixel of texture or depth is 8 bits.

# Complexity Assessment(6/8)

20

Number of operations						
PU_size	Proposal (Average)		Anchor		Ratio (to anchor)	
	Add/Sub/ Abs/Comp	Cons_Mul	Add/Sub/ Abs/Comp	Cons_Mul	Add/Sub/ Abs/Comp	Cons_Mul
64x64	34880	24576	32768	24576	106.45%	100.00%
64x32	17440	12288	16384	12288	106.45%	100.00%
32x64	17440	12288	16384	12288	106.45%	100.00%
32x32	8720	6144	8192	6144	106.45%	100.00%
32x16	4360	3072	4096	3072	106.45%	100.00%
16x32	4360	3072	4096	3072	106.45%	100.00%
16x16	2180	1536	2048	1536	106.45%	100.00%
16x8	1090	768	1024	768	106.45%	100.00%
8x16	1090	768	1024	768	106.45%	100.00%
8x8	545	384	512	384	106.45%	100.00%
8x4	256	192	256	192	100.00%	100.00%
4x8	256	192	256	192	100.00%	100.00%

# Complexity Assessment(7/8)

Data storage requirement (Bits)				
PU_size	Proposal (Average)	Anchor	Ratio (to anchor)	Difference bits (to anchor)
64x64	66668	65932	101.12%	736
64x32	33900	33164	102.22%	736
32x64	33900	33164	102.22%	736
32x32	17516	16780	104.39%	736
32x16	9324	8588	108.57%	736
16x32	9324	8588	108.57%	736
16x16	5228	4492	116.38%	736
16x8	3180	2444	130.11%	736
8x16	3180	2444	130.11%	736
8x8	2156	1420	151.83%	736
8x4	1132	908	124.67%	224
4x8	1388	908	152.86%	480
Average			119.42%	672

# Complexity Assessment(8/8)

Data transfer rate (Bits/PU)			
PU_size	Proposal (Average)	Anchor	Ratio (to anchor)
64x64	142007	163852	86.67%
64x32	71010	90124	78.79%
32x64	71010	90124	78.79%
32x32	35511	45068	78.79%
32x16	17762	22540	78.80%
16x32	17762	22540	78.80%
16x16	8887	11276	78.81%
16x8	4450	5644	78.84%
8x16	4450	5644	78.84%
8x8	2231	2828	78.89%
8x4	1036	1420	72.96%
4x8	1292	1420	90.99%
Average			80.00%

# Conclusion

- Coding Performance of method 4
  - 0.1% bitrate increasing for video PSNR vs. video bitrate
  - Slightly bitrate increasing video PSNR vs. total bitrate and synth PSNR vs. total bitrate in average
  - 0.2 % decoding time increasing
- Complexity of method 4 vs. current B-VSP
  - 6% Number of operations increasing
  - 20% data storage size increasing
    - Larger block size needed more reference texture data
    - Exactly increased size: about 480 ~ 736 bits
  - 20 % data transfer rate per PU decreasing
  - Number of data fetching from decoded picture buffer is reduced
- Adopted method 4 and further studied on larger sub-block.



Thanks for ITRI's  
crosscheck (JCT3V-D0292)