

JCT3V-D0165

CE1.h: VSP Complexity Analysis and Constrained VSP (CVSP)

Feng Zou, Dong Tian, Anthony Vetro, Huifang Sun
Mitsubishi Electric Research Labs (MERL)
Cambridge, MA

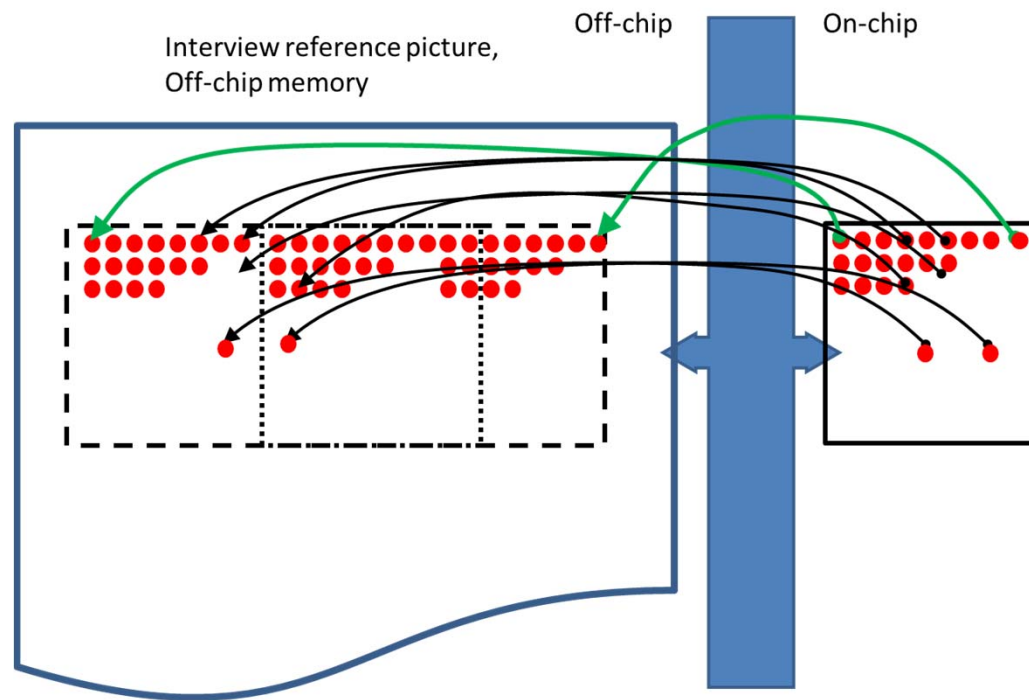
4th JCT-3V meeting, Incheon, KR, April 2013

Assumptions for HW implementations

- This contribution
 - Study both off-chip and on-chip complexity
 - Propose constraints to simplify VSP implementation
- DPB buffer stores in off-chip memory (SDRAM)
- HEVC interpolation process
 - 8- or 7- tap filters
 - On-chip processing elements (PE)
 - Supported by on-chip local memory (LM)
- Prediction block (PB) is represented as $M \times N$
 - M: Width
 - N: Height

Off-chip memory access

- VSP imposes irregular addressing
- Also, off-chip memory bandwidth increased!
 - 1/3 more data fetching
 - 8 times addressing



Proposal: Constrained VSP (CVSP)

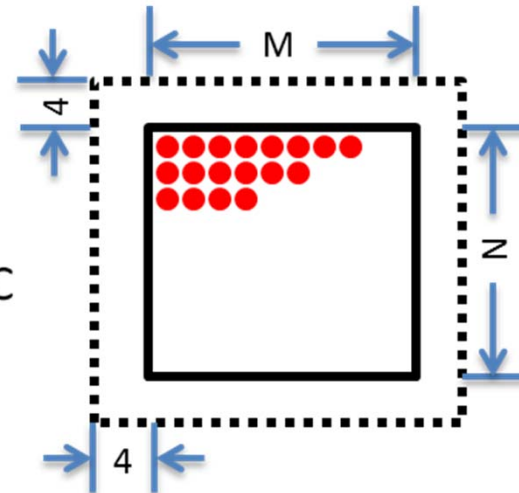
Constrain VSP to maintain the same amount of pixel fetching from DPB as traditional MC

$$(M+8) \times (N+8)$$

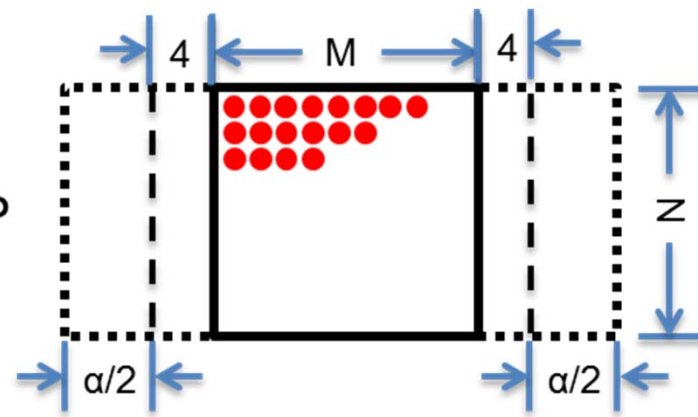


$$(M+\alpha+8) \times N$$

Traditional MC

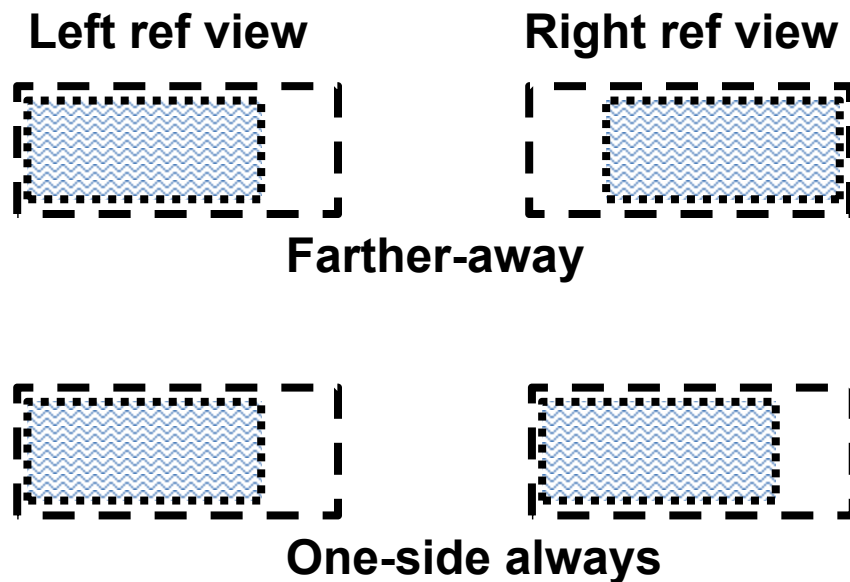


1x1 CVSP



Proposed CVSP – apply constraints on depth

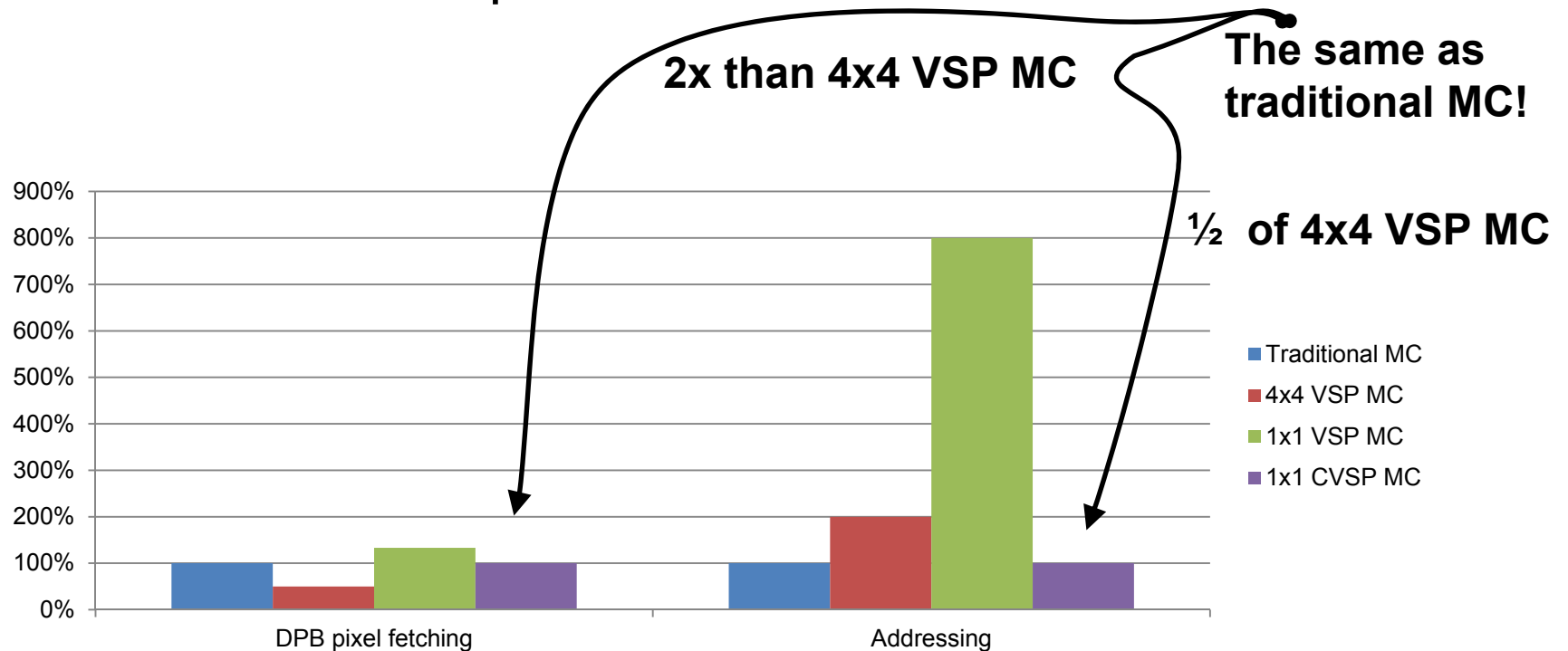
- Get the window width of the reference pixels
 - Determine α to keep the number of pixels not increased
- Get a DV for the reference pixels at PU level
 - Method 1: Farther-away alignment
 - Method 2: Left-side alignment



M	N	Traditional MC (M+8) x (N+8)	CVSP	α	M+ α
			(M+ α + 8) x N		
4	8	192	192	12	16
8	4	192	192	32	40
8	8	256	256	16	24
8	16	384	384	8	16
16	8	384	384	24	40
16	16	576	576	12	28
16	32	960	960	6	22
32	16	960	960	20	52
32	32	1600	1600	10	42
32	64	2880	2880	5	37
64	32	2880	2880	18	82
64	64	5184	5184	9	73

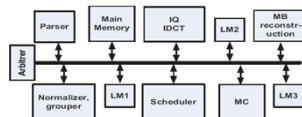
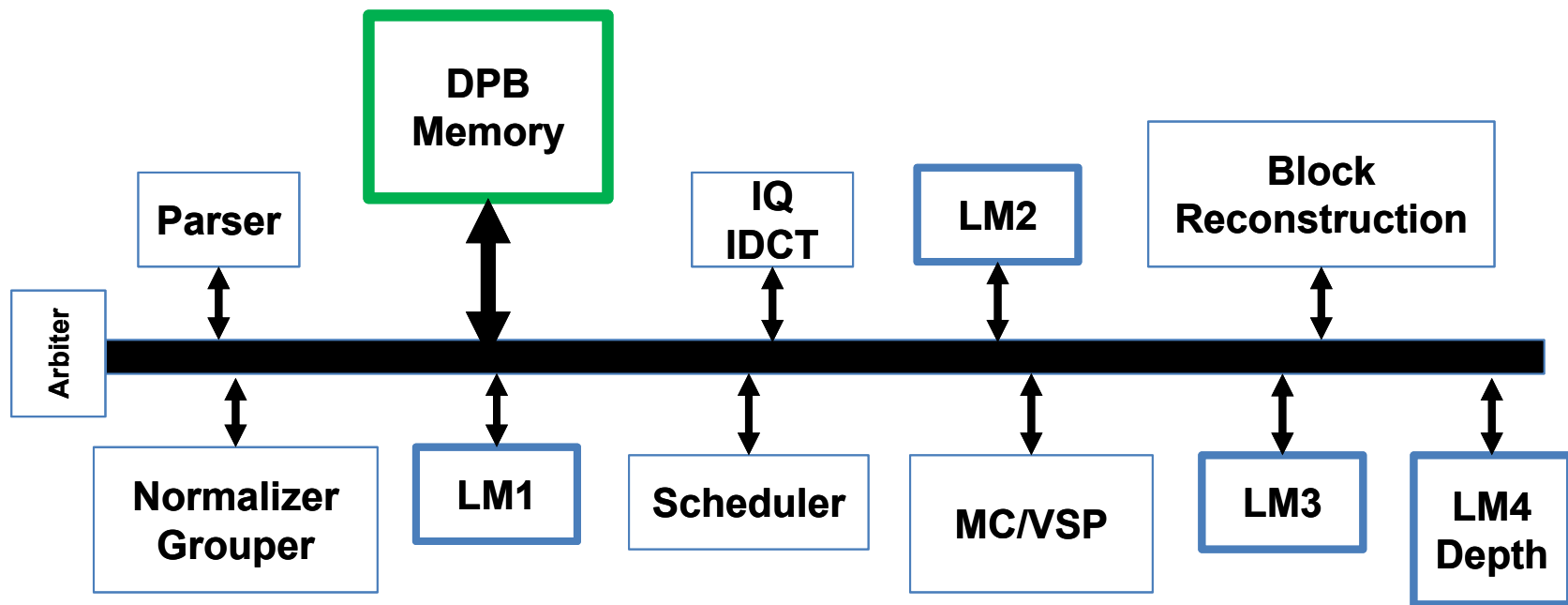
Off-chip memory access – same as traditional MC

- The proposed CVSP achieves the same off-chip complexity!
 - Amount of DPB pixel fetching
 - Addressing times
 - DV for reference pixels can be determined at PU level



HW block diagram for a decoder

- Block diagram of a typical decoder design
 - Modified to incorporate depth



H. Habli, et.al., Optimizing Off-Chip Memory Access Costs in Low Power MPEG-4 Decoder, *IC/CS'12*, April 3–5, 2012,

Modified /Added blocks

- MC module
 - Modified to support both traditional MC and VSP MC
- New local memory*
 - LM4 for depth, ($\leq 4K$, 64×64 , or $64 +$ memory reuse)
- Convert the depth values in LM4 to disparity values*
- Convert the disparities to memory vector
 - May re-use LM4 memory
- Linear memory reading from DPB with proposed CVSP
 - Processing memory vectors in LM4*
 - Similar to processing memory vectors in LM1

* Shared with DoNBDV procedure

High level factors

- Total number of interpolation
 - Computation
 - On-chip memory bandwidth
- On-chip pixel loading
 - On-chip memory bandwidth
- DPB pixel fetching
 - Off-chip memory bandwidth
- DPB addressing
 - Computation

**On-chip
complexity**

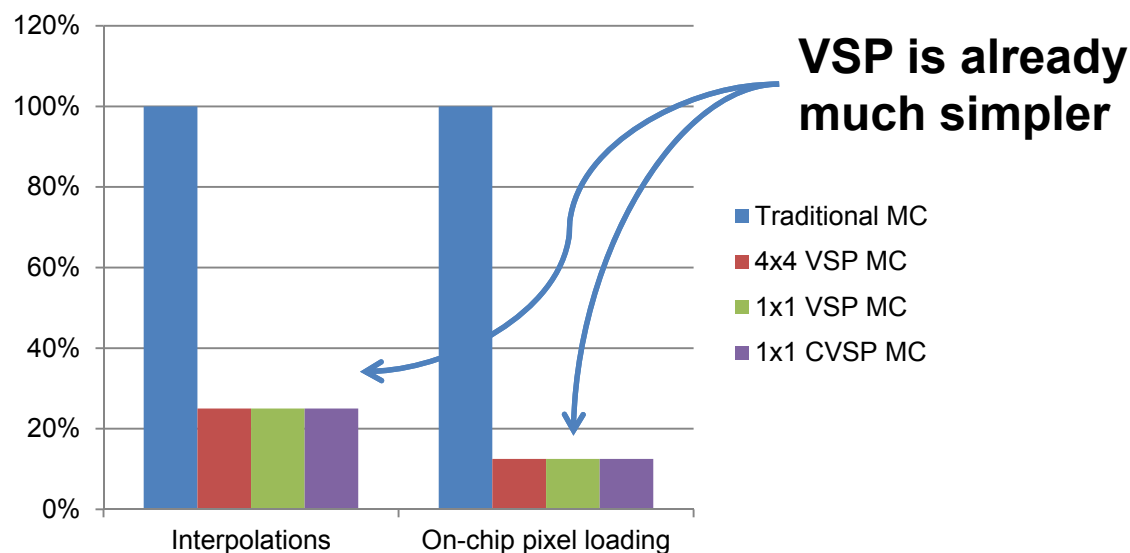
**Off-chip
complexity**

On a 64x64 block with all 8x4 PB partitions

On-chip complexity

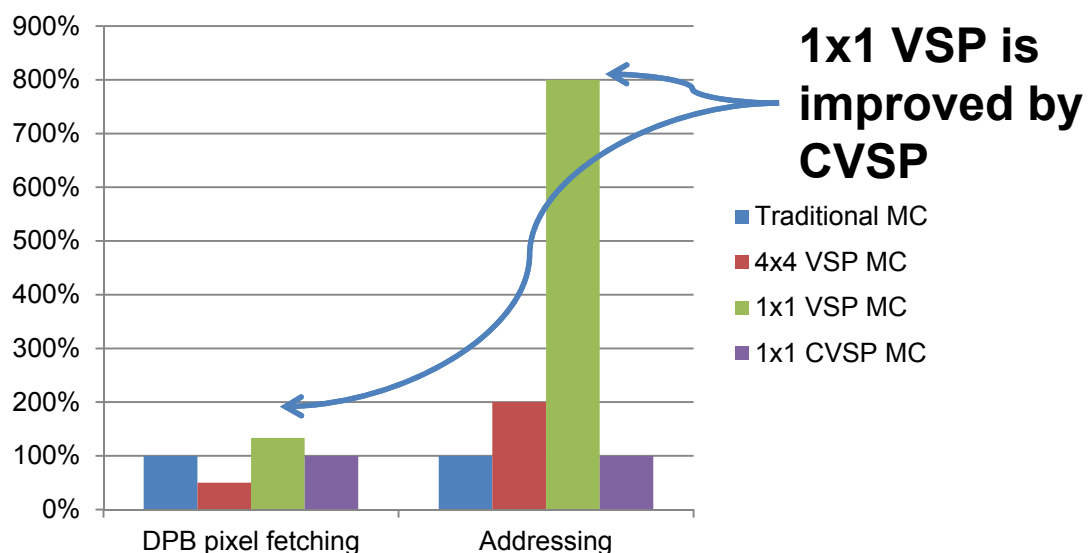
- **Memory bandwidth**
- **Computation**

Gap for on-chip pixel loading may be closer in data reusing case



Off-chip complexity

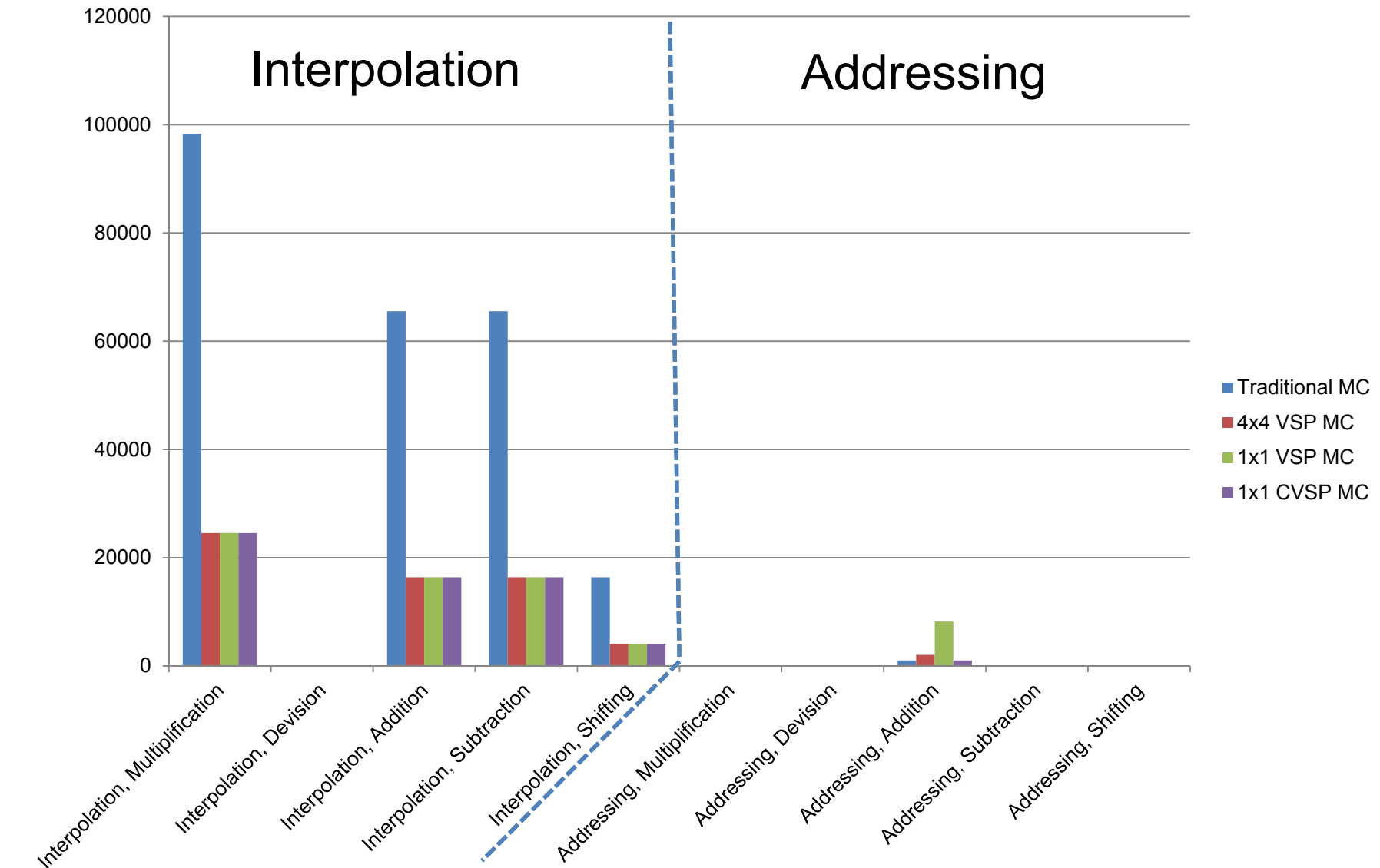
- **Memory bandwidth**
- **Computation**



Low level factors for arithmetic calculations

- Factors
 - Multiplication
 - Division
 - Addition
 - Subtraction
 - Shifting
- Arithmetic calculations
 - Interpolation
 - Address calculation

On a 64x64 block with all 8x4 PB partitions



Simulations on CVSP

- Results of “further-away” method
 - Overall bitrate saving is 0.2%
 - Almost without coding time changes

	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%	-0.2%	0.1%	0.0%	0.0%	-0.1%	100.3%	95.5%	100.1%
Kendo	0.0%	-0.1%	0.0%	0.0%	-0.1%	0.0%	100.8%	96.0%	100.4%
Newspaper_CC	0.0%	-0.2%	-0.1%	0.0%	0.0%	-0.1%	100.6%	93.2%	100.2%
GT_Fly	0.0%	-0.3%	-0.4%	-0.1%	-0.1%	-0.1%	98.8%	101.4%	97.8%
Poznan_Hall2	0.0%	-0.4%	0.1%	-0.1%	-0.1%	0.0%	99.5%	103.5%	99.5%
Poznan_Street	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	99.9%	113.7%	95.4%
Undo_Dancer	0.0%	-3.9%	-3.1%	-1.0%	-1.0%	-1.1%	98.0%	100.9%	97.7%
1024x768	0.0%	-0.1%	0.0%	0.0%	0.0%	-0.1%	100.6%	94.9%	100.2%
1920x1088	0.0%	-1.1%	-0.9%	-0.3%	-0.3%	-0.3%	99.0%	104.9%	97.6%
average	0.0%	-0.7%	-0.5%	-0.2%	-0.2%	-0.2%	99.7%	100.6%	98.7%

1x1 VSP performance, well underestimated

- Though a full synthetic picture looks satisfactory, the use of VSP pixels still very low, only ~3%!
- In concept, VSP should be a very powerful tool to exploit interview redundancy
 - Why not more pixels VSP skipped? ~30%? ...



**Synthetic pictures
@ QP 40**

- Subjective measurements should be given high priority

Summary of 1x1 CVSP

Complexity Factors	1x1 CVSP MC vs. Traditional MC
Off-chip memory bandwidth	<u>Same as benchmark</u>
Off-chip memory access	<u>Random access, line memory reading, no irregular addressing problem, same as benchmark</u>
On-chip memory bandwidth	~1/8 or (1/4) of benchmark*
On-chip computation complexity	~1/8 or (1/4) of benchmark
Additional on-chip local memory	Store a depth block, but shared with NBDV process
Encoder side!	No ME at encoder!
Performance	Higher gain achieved (0.2%), more potential expected

Conclusions

- 1x1 VSP MC vs. Traditional MC
 - Off-chip complexity: 1x1 VSP is more complex
 - On-chip complexity: 1x1 VSP is less complex or similar
- 1x1 CVSP MC vs. Traditional MC
 - 1x1 CVSP is much less complex
- 1x1 CVSP MC vs. 4x4 VSP MC
 - Requires $\frac{1}{2}$ off-chip addressing, 2x off-chip pixels loading
 - On-chip pixel access a bit complex, but acceptable
 - Overall, similar complexity as 4x4 VSP MC
- 1x1 CVSP MC maintains the performance of 1x1 VSP MC
- Recommend to adopt 1x1 CVSP
 - Higher performance with significantly reduced complexity

THANKS

Especially to
NTT, JCT3V-D0225
& MediaTek, JCT3V-D0293
for their valuable time in cross checking

Complexity comparison on an MxN block

Decoder Side Analysis	Traditional MC	4x4 VSP MC	1x1 VSP MC	Proposed CVSP MC (1x1)
A: Number of fetched pixels from DPB	$(M+8) \times (N+8)$	$4 \times (4+8) \times M \times N / (4 \times 4)$	$M \times N \times 8$	$(M+8+\alpha) \times N = (M+8) \times (N+8)$, α is the maximum disparity constraint
B: Addressing calculation	N (line buffer read)	$N \times (M/4)$ (line buffer read)	$M \times N$	N (line buffer read)
C: Maximum total number of interpolation	$(8+1) \times M \times N$ (not consider data reuse) $(N+8) \times M + N \times M$ $= (2N+8) \times M$ (consider data reuse)	$M \times N$,	$M \times N$	$M \times N$ (interpolation used only for horizontal direction)
D: Number of read out pixels	$8 \times 8 \times M \times N = 64 \times M \times N$	$8 \times M \times N$	$8 \times M \times N$	$8 \times M \times N$

On a 64x64 block with all 8x4 PB partitions

High level

Decoder Side Analysis	Traditional MC	4x4 VSP MC	1x1 VSP MC	Proposed CVSP MC (1x1)
A: Number of fetched pixels from DPB	192*128=24,576	96*128=12,288 1/2	256*128=32,768 1/3 more	192*128=24,576 1x
B: Addressing calculation	4*128=512 (line buffer read)	8*128=1,024 (line buffer read) 2x	32*128=4,096 8x	4*128=512 (line buffer read) 1x
C: Maximum total number of interpolation	288*128=36,864 (not consider data reuse) 128*128=16,384 (consider vertical interpolated intermediate fractional data reuse)	32*128=4,096 ~1/8 or 1/4	32*128=4,096 ~1/8 or 1/4	32*128=4,096 (interpolation used only for horizontal direction) ~1/8 or 1/4
D: Number of read out pixels	2,048*128=262,144	256*128=32,768 1/8	256*128=32,768 1/8	256*128=32,768 1/8

On a 64x64 block with all 8x4 PB partitions

Low level

		Traditional MC	4x4 VSP MC	1x1 VSP MC	Proposed CVSP MC (1x1)
Interpolation	X	$768 * 128 = 98,304$	$192 * 128 = 24,576$	$192 * 128 = 24,576$	$192 * 128 = 24,576$
	/	0	0	0	0
	+	$512 * 128 = 65,536$	$128 * 128 = 16,384$	$128 * 128 = 16,384$	$128 * 128 = 16,384$
	-	$512 * 128 = 65,536$	$128 * 128 = 16,384$	$128 * 128 = 16,384$	$128 * 128 = 16,384$
	>>	$128 * 128 = 16,384$	$32 * 128 = 4,096$	$32 * 128 = 4,096$	$32 * 128 = 4,096$
Addressing calculation	X	0	0	0	0
	/	0	0	0	0
	+	$8 * 128 = 1,024$	$16 * 128 = 2,048$	$64 * 128 = 8,192$	$8 * 128 = 1,024$
	-	0	0	0	0
	>>	0	0	0	0

About LM4, 4K or 64

- In NBDV, 64 bytes required, it is not necessary to buffer the whole depth block
- NBDV returns a DV from the fetched depth block
- CVSP returns a DV from the fetched depth block as well
 - Used for DPB pixel fetching
 - Indicating the starting address
- The above two processes can be harmonized
 - Can be line based processing
- The fetched depth line is required for MC
 - Can be line based processing
- If the 4K is allowed, one time depth fetching
- Otherwise, two times depth fetching ($64 \times 64 / 8 / 4 \times 2 = 256$ LM1)

On-chip pixel loading

- On-chip pixel loading without data reuse
 - 1x1 VSP MC is 1/8 of traditional MC
- On-chip pixel loading with data reuse
 - 8x4 traditional MC: 15 pixels/line * (4 lines + 8 padded lines) = 180 + data reuse logic
 - 4x4 VSP MC: 11 pixels/line * (4 lines + 0 padded lines) * 2 block = 88 + data reuse logic
 - 1x1 VSP MC: 8 pixels * 8 columns * 4 lines = 256 w/o data reuse logic
- 1x1 VSP MC comparable to traditional MC
- 1x1 VSP MC a bit higher than 4x4 VSP MC

Summary of on-chip complexity

64x64 block w/ 8x4 partitions

Complexity Factors	1x1 CVSP MC vs. Traditional MC
Interpolations	25% (4,096 vs. 16,384)
On chip pixel loading (8x4)	256 vs. 180 + logics for reusing
LM4 memory	4096 vs. none; or 64 vs. none
Depth/MV loading amount	64x64=4096, vs. 256
Depth loading times	64 vs. none

**Extra, but may be shared with
other procedures**