CE2: ON RESTRICTION OF MEMORY BANDWIDTH CONSUMPTION OF AFFINE MODE (CE2-4.8)

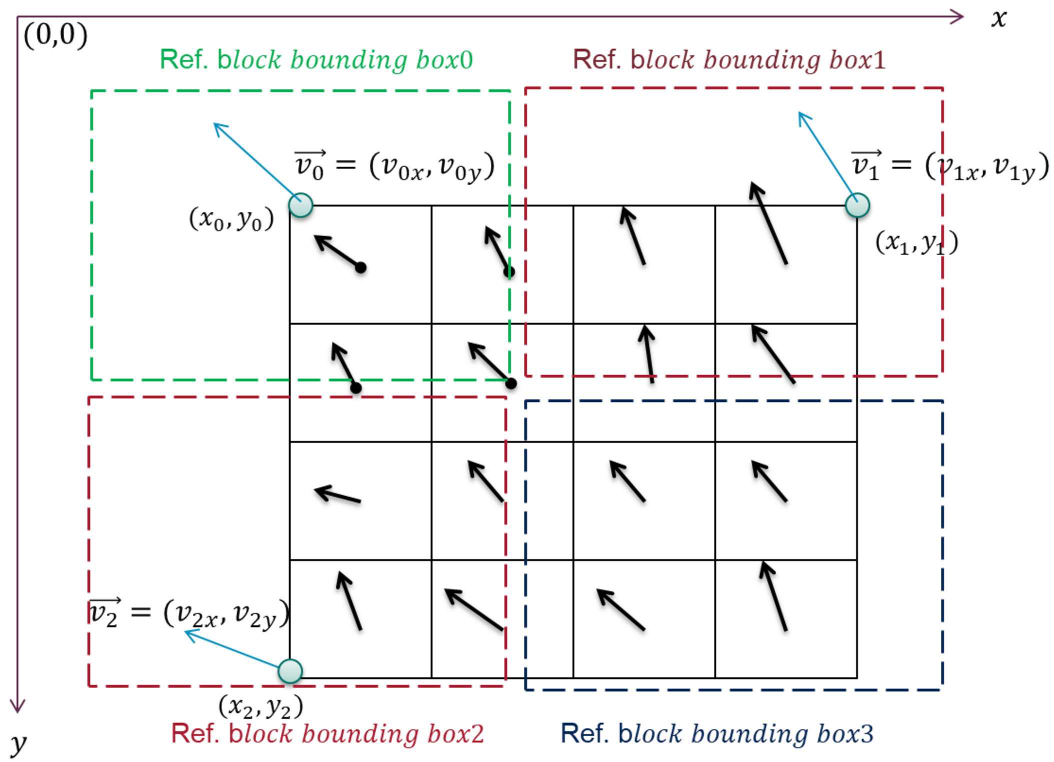
(JVET-N0068)

Minhua Zhou

Broadcom Inc.

Background

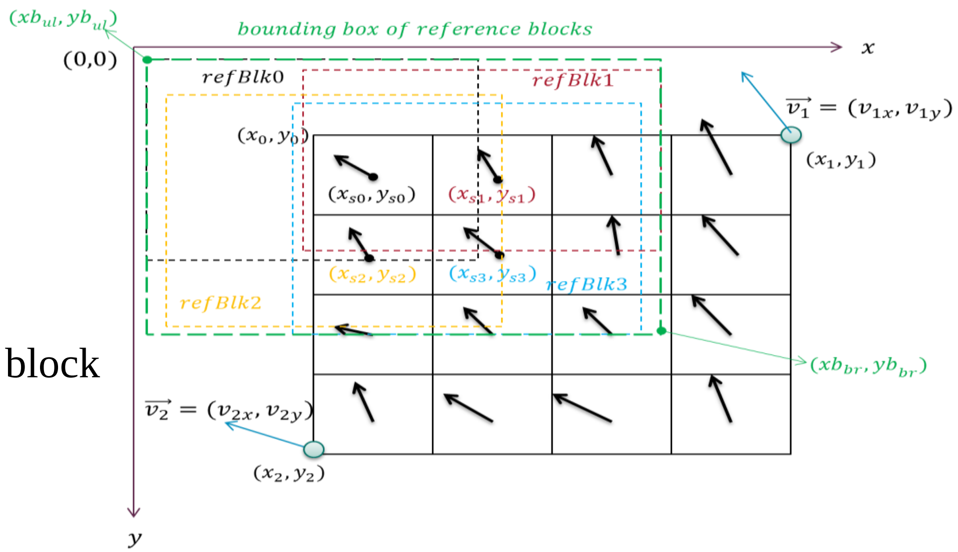
- In the current design, 4x4 sub-block size is used for the generation of sub-block vectors of affine mode
- The sub-block motion vector spread (i.e. how far the sub-block motion vectors can go apart from each other) needs to be constrained to avoid non-affordable peak memory bandwidth consumption of the affine mode.
- Most of methods being studied just constrain the sub-block motion vector spread for each 8x8 region of a PU
  - Within the PU the sub-block motion vectors can still go widely apart from one 8x8 region to the next 8x8 region
- Propose to impose the constraint at the PU level, not just at 8x8 level.

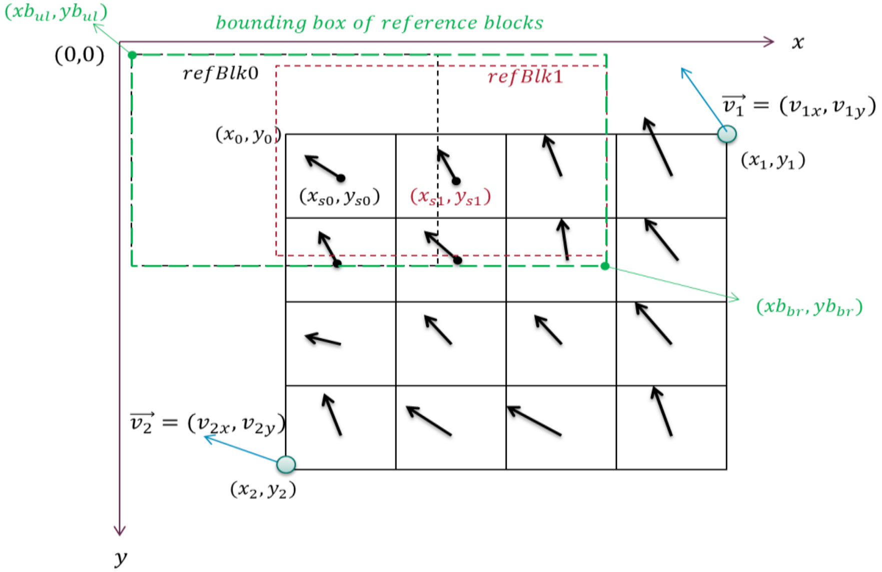


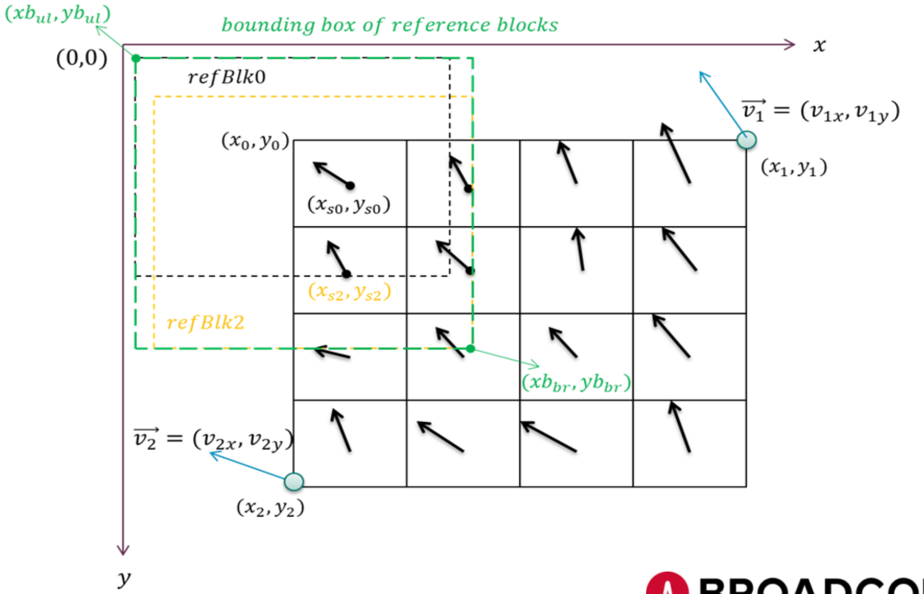
Proposed Method

- Directly constrain the affine motion model parameters
- If the following conditions are satisfied, the regular affine MC applies
  - If a PU uses bi-directional affine mode (i.e. bi-pred), the reference block bounding box size $bxW4 * bxH4$ is less than or equal to $Thred4$.
  - Otherwise, if the PU is coded in unidirectional affine mode, the reference block bounding box size $bxW_h * bxH_h$ is less than or equal to $Thred_h$ and the reference block bounding box size $bxW_v * bxH_v$ is less than or equal to $Thred_v$.

Otherwise, all the sub-block vectors of the PU is set to a same one which is generated at the center point of the PU by using the affine motion model, single vector is used for the MC of affine coded PU.







Proposed Method-Cont.

- The affine motion model
  - $$\begin{cases} v_x = ax + cy + e \\ v_y = bx + dy + f \end{cases}$$
- The reference block bounding boxes are computed by:
  - $$\begin{cases} bxW4 = \max(0, 4(1+a), 4c, 4(1+a)+4c) - \min(0, 4(1+a), 4c, 4(1+a)+4c) + 11 \\ bxH4 = \max\big(0, 4b, 4(1+d), 4b+4(1+d)\big) - \min\big(0, 4b, 4(1+d), 4b+4(1+d)\big) + 11 \end{cases}$$
  - $$\begin{cases} bxW_h = \max\big(0, 4(1+a)\big) - \min\big(0, 4(1+a)\big) + 11 \\ bxH_h = \max(0, 4b) - \min(0, 4b) + 11 \end{cases}$$
  - $$\begin{cases} bxW_v = \max(0, 4c) - \min(0, 4c) + 11 \\ bxH_v = \max(0, 4(1+d)) - \min(0, 4(1+d)) + 11 \end{cases}$$
- Pre-defined thresholds

$$\begin{cases} Thred4 = (15+\delta_x) * (15+\delta_y) \\ Thred_h = (15+\delta_x) * (11+\delta_y) \\ Thred_v = (11+\delta_x) * (15+\delta_y) \end{cases}$$

- If the following conditions are satisfied, the regular affine MC is used. Otherwise, the fallback mode is triggered.

$$\begin{cases} bxW_4 * bxH_4 \le Thred_4 & for\ bi-directional\ affine\ mode \\ bxW_h * bxH_h \le Thred_h\ \ and\ \ bxW_v * bxH_v \le Thred_v & for\ unidirectional\ affine\ mode \end{cases}$$

Experimental Results

- Tests carried out ($setting\ \delta_x = \delta_y = 2$)
  1) Test CE2-4.8a : the encoder controls the affine motion estimation so that the fallback mode is never triggered in the bitstreams.
  2) Test CE2-4.8b : the encoder allows the use of fallback mode if the reference block bounding box sizes exceed the pre-defined thresholds.
- Summary results:

Test 1

	Random Access Main 10				
	Over VTM4.0				
	Y	U	V	EncT	DecT
Class A1	0.02%	-0.01%	0.06%	90%	93%
Class A2	0.05%	0.02%	0.05%	101%	105%
Class B	0.02%	-0.02%	0.06%	89%	89%
Class C	0.01%	0.00%	-0.03%	105%	106%
Class E					
Overall	0.02%	0.00%	0.03%	95%	97%
Class D	0.01%	0.02%	-0.06%	95%	102%
Class F (optional)	0.05%	0.07%	0.03%	94%	101%

	Low delay B Main10				
	Over VTM4.0				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	0.01%	-0.02%	-0.16%	98%	98%
Class C	-0.04%	0.26%	0.13%	100%	95%
Class E	0.02%	0.39%	-0.67%	73%	85%
Overall	0.00%	0.18%	-0.19%	92%	94%
Class D	-0.05%	-0.33%	-0.48%	93%	91%
Class F (optional)	-0.16%	-0.17%	-0.72%	91%	85%

Test 2

	Random Access Main 10				
	Over VTM4.0				
	Y	U	V	EncT	DecT
Class A1	0.01%	0.03%	0.03%	104%	104%
Class A2	-0.01%	0.00%	0.01%	101%	97%
Class B	0.00%	0.01%	-0.02%	98%	97%
Class C	-0.02%	-0.07%	0.02%	93%	100%
Class E					
Overall	-0.01%	-0.01%	0.01%	98%	99%
Class D	-0.01%	-0.11%	-0.03%	95%	103%
Class F (optional)	-0.01%	0.03%	-0.04%	91%	97%

	Low delay B Main10				
	Over VTM4.0				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	0.00%	0.02%	-0.20%	111%	107%
Class C	-0.02%	0.34%	-0.23%	105%	100%
Class E	0.04%	0.18%	-0.40%	86%	90%
Overall	0.01%	0.17%	-0.26%	102%	100%
Class D	0.00%	-0.08%	-0.67%	106%	107%
Class F (optional)	-0.09%	-0.31%	-0.58%	105%	95%

Thanks to ByteDance for cross-check

Memory Bandwidth Measurements (RA)

- Memory bandwidth measured by using a commercial MC cache model.
- No memory bandwidth impact has been observed by imposing the restrictions.
  - TCM_diff : Total cache misses (over all the frames coded), percentage increase relative to VTM4.0.
  - ABW_diff: Average memory bandwidth (over all the frames coded), percentage increase relative to VTM4.0.
  - MBW_diff : Worst case (Max) memory bandwidth (among all the frames coded), percentage increase relative to VTM4.0.

	Random Access Main 10							
	Over VTM-4.0							
	Y	U	V	TCM_diff	ABW_diff	MBW_diff	EncT	DecT
Class A1	0.02%	-0.01%	0.06%	-0.06%	-0.11%	-0.32%	90%	108%
Class A2	0.05%	0.02%	0.05%	-0.02%	-0.16%	0.32%	101%	115%
Class B	0.02%	-0.02%	0.06%	-0.02%	-0.04%	0.00%	89%	116%
Class C	0.01%	0.00%	-0.03%	0.02%	0.02%	0.46%	105%	113%
Class E								
Overall	0.02%	0.00%	0.03%	-0.03%	-0.05%	0.46%	95%	113%
Class D	0.01%	0.02%	-0.06%	-0.03%	0.00%	-0.21%	95%	114%
Class F (optional)	0.05%	0.07%	0.03%	-0.04%	0.00%	0.31%	94%	115%

Table 4: Memory bandwidth measurement for Test CE2-4.8a $(\delta_x = \delta_y = 2)$

	Random Access Main 10							
	Over VTM-4.0							
	Y	U	V	TCM_diff	ABW_diff	MBW_diff	EncT	DecT
Class A1	0.01%	0.03%	0.03%	-0.04%	-0.04%	-0.32%	104%	99%
Class A2	-0.01%	0.00%	0.01%	0.03%	-0.03%	0.32%	101%	124%
Class B	0.00%	0.01%	-0.02%	-0.02%	0.02%	-0.56%	98%	109%
Class C	-0.02%	-0.07%	0.02%	0.04%	0.11%	0.70%	93%	117%
Class E								
Overall	-0.01%	-0.01%	0.01%	0.01%	0.03%	0.70%	98%	112%
Class D	-0.01%	-0.11%	-0.03%	-0.03%	0.02%	-0.21%	95%	121%
Class F (optional)	-0.01%	0.03%	-0.04%	0.08%	0.08%	0.62%	91%	112%

Table 5: Memory bandwidth measurement for Test CE2-4.8b $(\delta_x = \delta_y = 2)$