

M31173/JCT3V-F0162

# 3D-CE1.h related: Vertical Filtering for View Synthesis Prediction

Erh-Chung Ke

Ching-Chieh Lin

Jih-Sheng Tu

**Industrial Technology Research Institute**

Geneva Oct. 2013

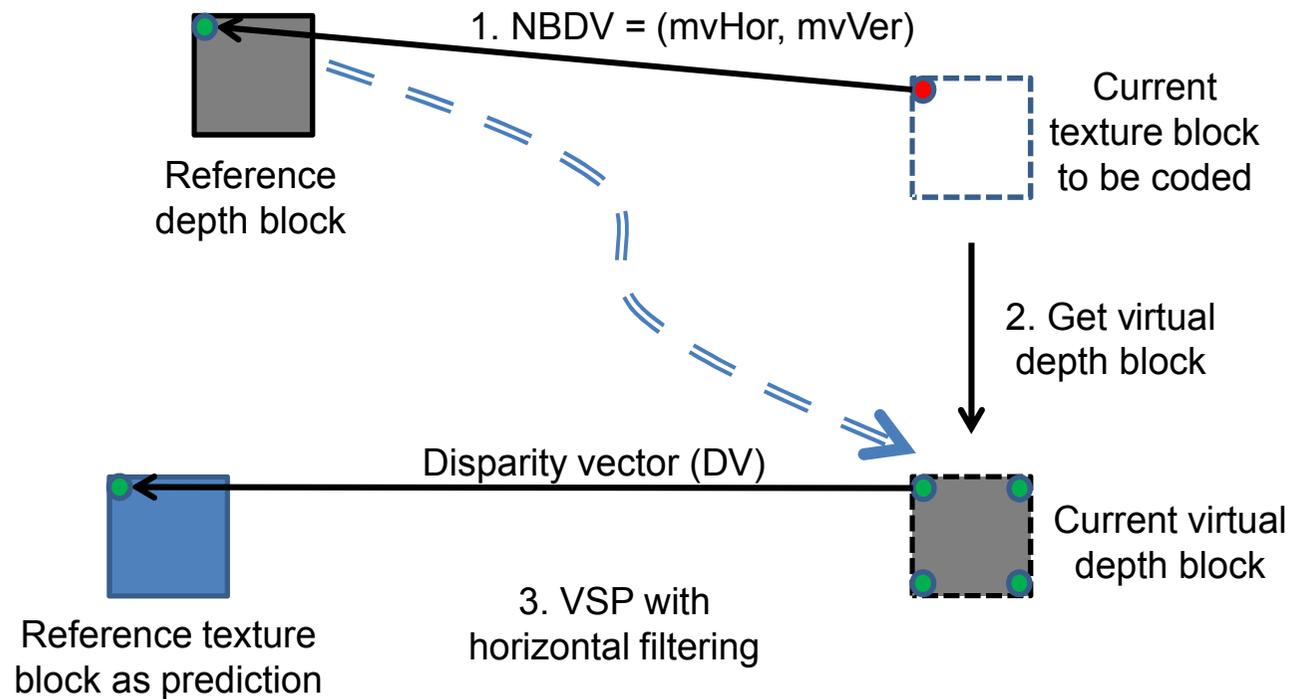
# Outline

- Current VSP
- Our solution for VSP
- Experimental results
- Conclusions

# Procedures of current VSP

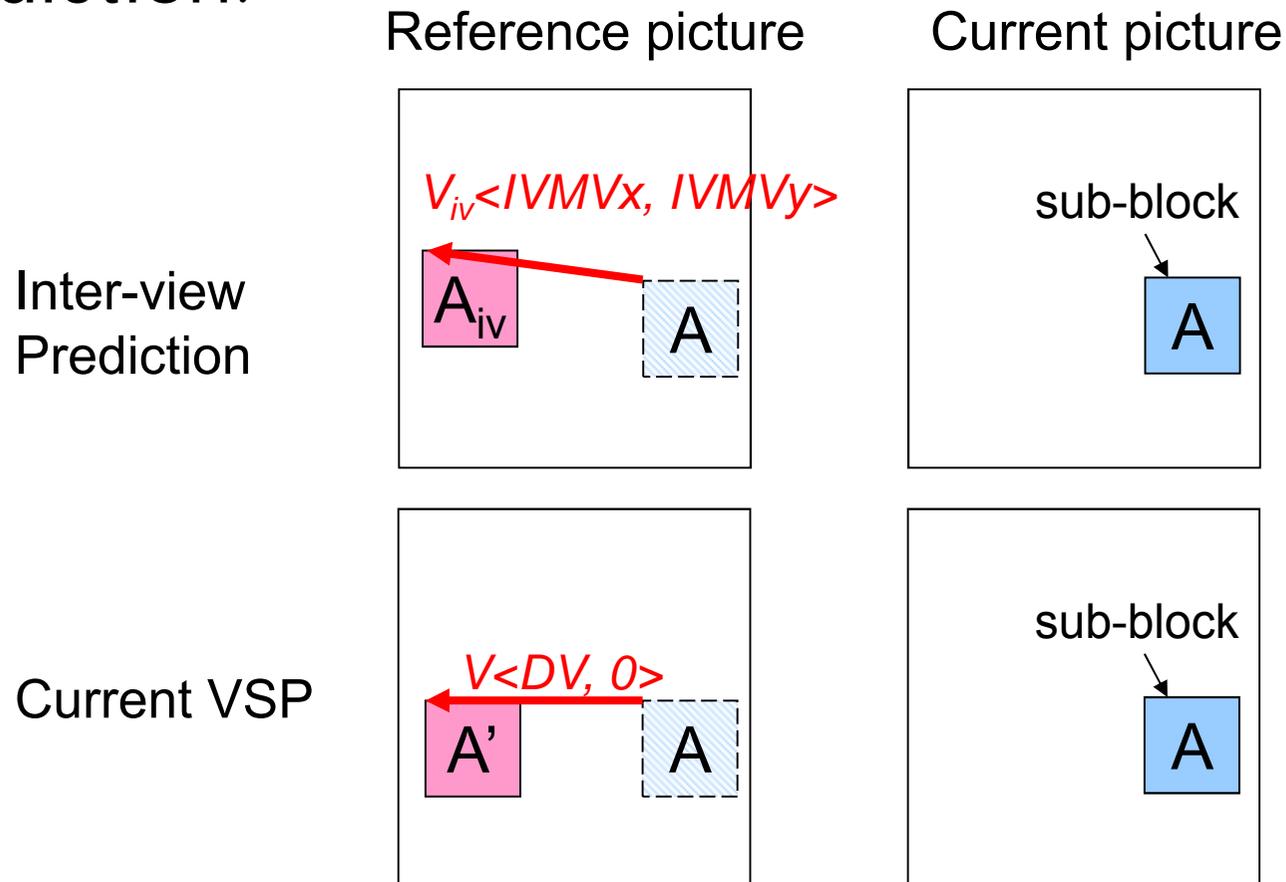
Reference view

Current view



# Issue with the current VSP

- VSP could be assumed as a special inter-view prediction.



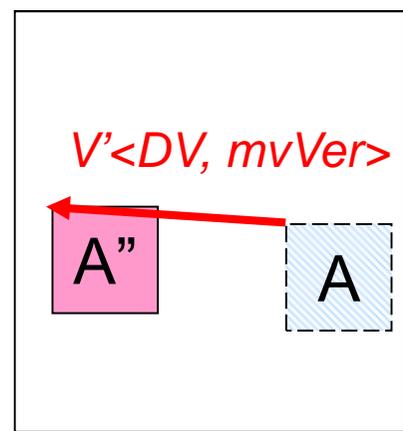
# ITRI's solution for VSP

- Propose a View Synthesis Motion Vector (VSMV)
  - The horizontal part of VSMV is identical to disparity vector.
  - The vertical part of VSMV will be set according to our proposed condition judgment.
- Apply VSMV when performing VSP.

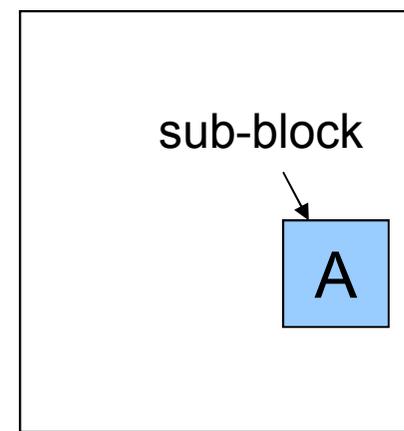
# Assignments of vertical part of VSMV

- If the vertical part of NBDV is in the range of an integer pixel, set the vertical part of VSMV as the vertical part of NBDV( $mvVer$ ).
- Otherwise, set the vertical part of VSMV as zero.

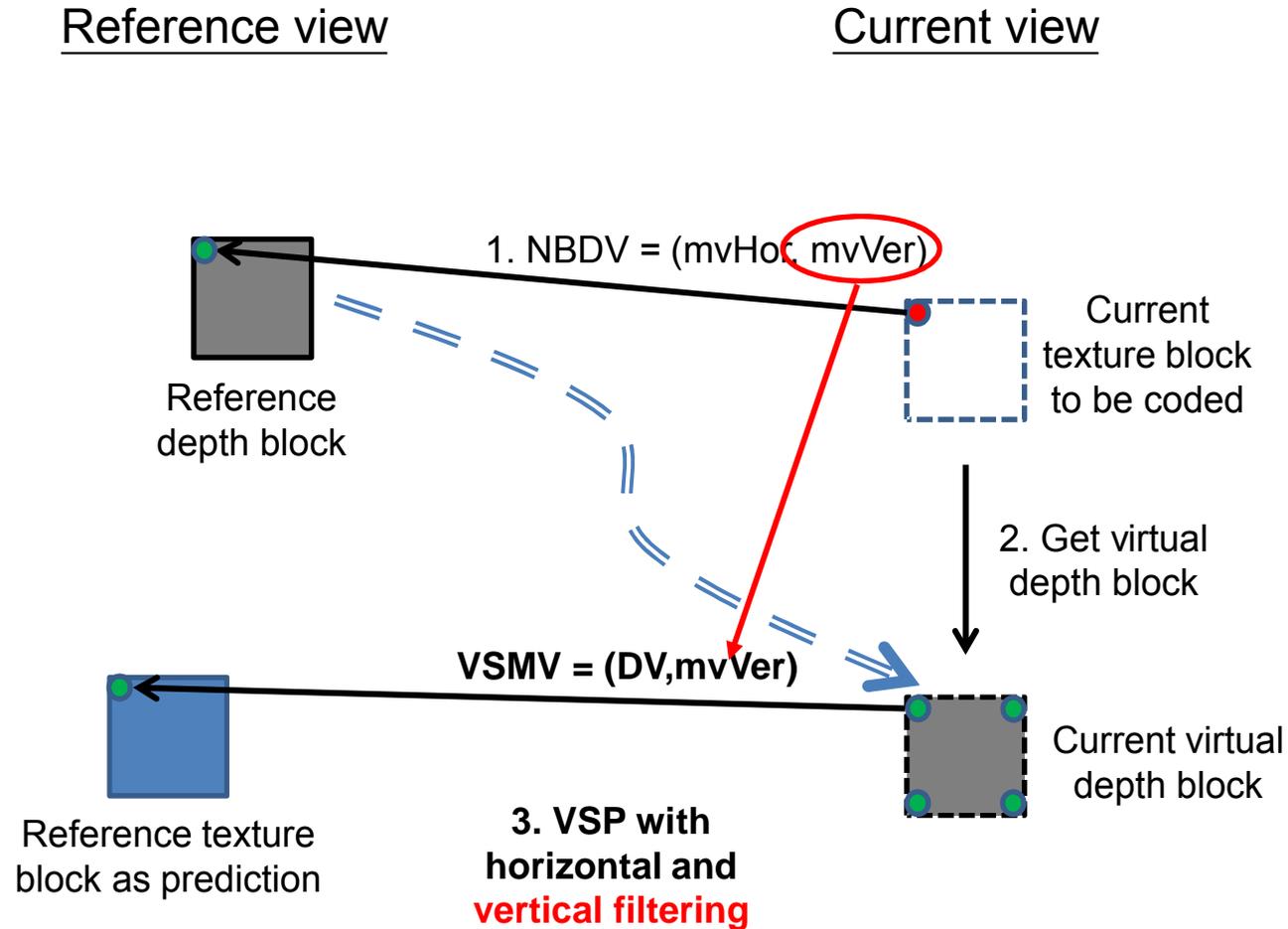
Reference picture



Current picture



# Procedures of our proposed VSP



# Experimental results

Compare to HTM 8.0

	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%	0.0%	100.8%	99.2%	101.0%
Kendo	0.0%	0.0%	-0.2%	0.0%	-0.1%	0.0%	101.3%	99.4%	100.6%
Newspaper_CC	0.0%	0.1%	0.1%	0.0%	0.0%	0.1%	101.4%	99.0%	88.6%
GT_Fly	0.0%	0.2%	0.1%	0.0%	0.0%	0.0%	100.3%	99.2%	100.9%
Poznan_Hall2	0.0%	-0.1%	-0.3%	-0.1%	-0.1%	-0.2%	96.8%	98.8%	95.7%
Poznan_Street	0.0%	-0.9%	-0.9%	-0.3%	-0.3%	-0.2%	98.3%	98.7%	100.0%
Undo_Dancer	0.0%	0.2%	0.1%	0.0%	0.0%	0.0%	99.2%	98.9%	103.2%
1024x768	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	101.2%	99.2%	96.7%
1920x1088	0.0%	-0.2%	-0.2%	-0.1%	-0.1%	-0.1%	98.7%	98.9%	99.9%
<b>average</b>	<b>0.0%</b>	<b>-0.1%</b>	<b>-0.2%</b>	<b>-0.1%</b>	<b>-0.1%</b>	<b>-0.1%</b>	<b>99.7%</b>	<b>99.0%</b>	<b>98.6%</b>
Shark	0.0%	-0.1%	0.2%	0.0%	0.0%	0.0%	107.4%	99.3%	103.0%

# Conclusions

- There is 0.1% gain for coded and synthesis view.
- Complexity is unchanged.

# Recommendation

- Recommend to adopt this proposal for 3D-HEVC

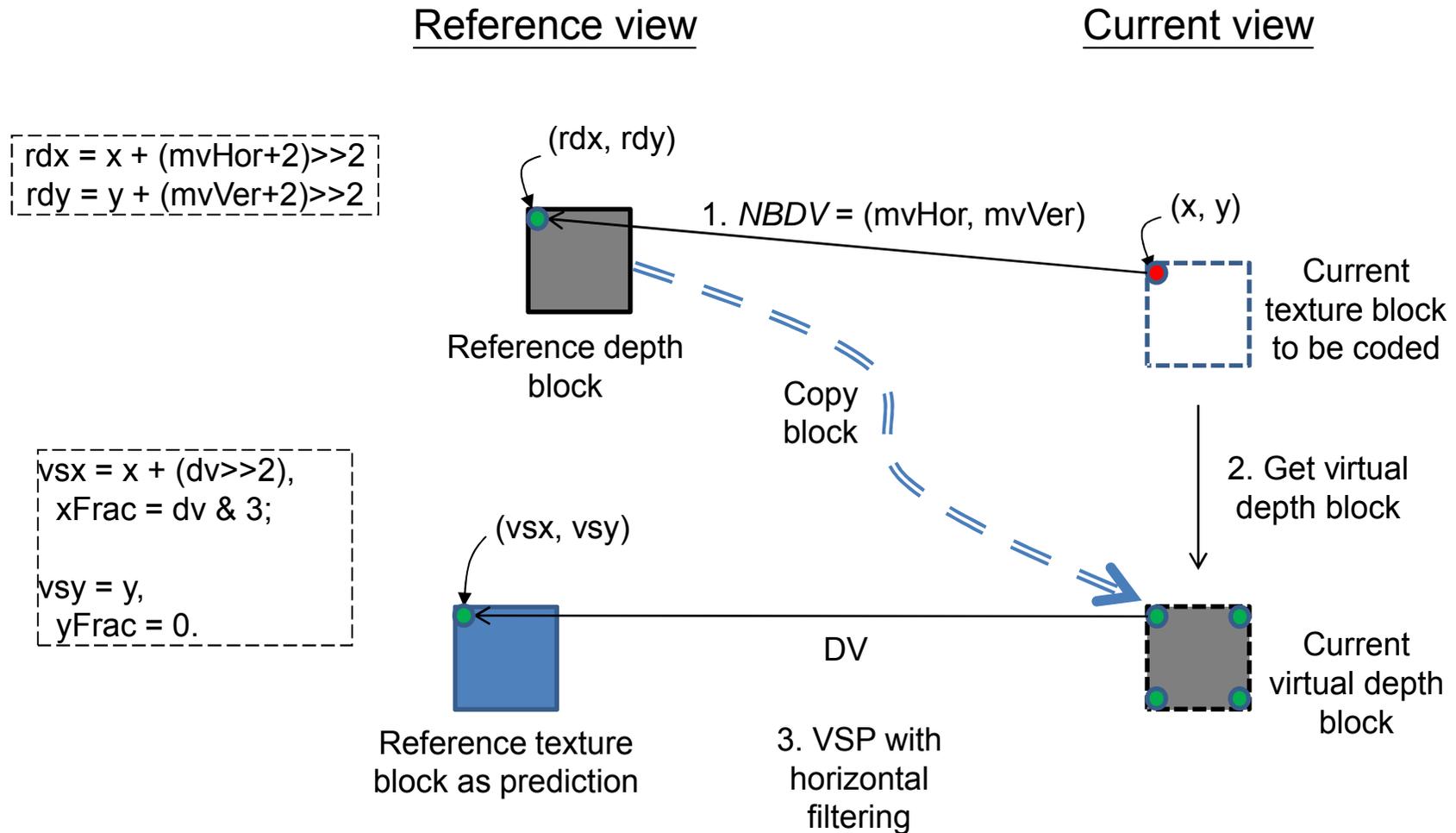
# Acknowledgment

- Thanks MediaTek for cross check

# Thank You !

# Appendix

# Detailed procedure of the current VSP in luma



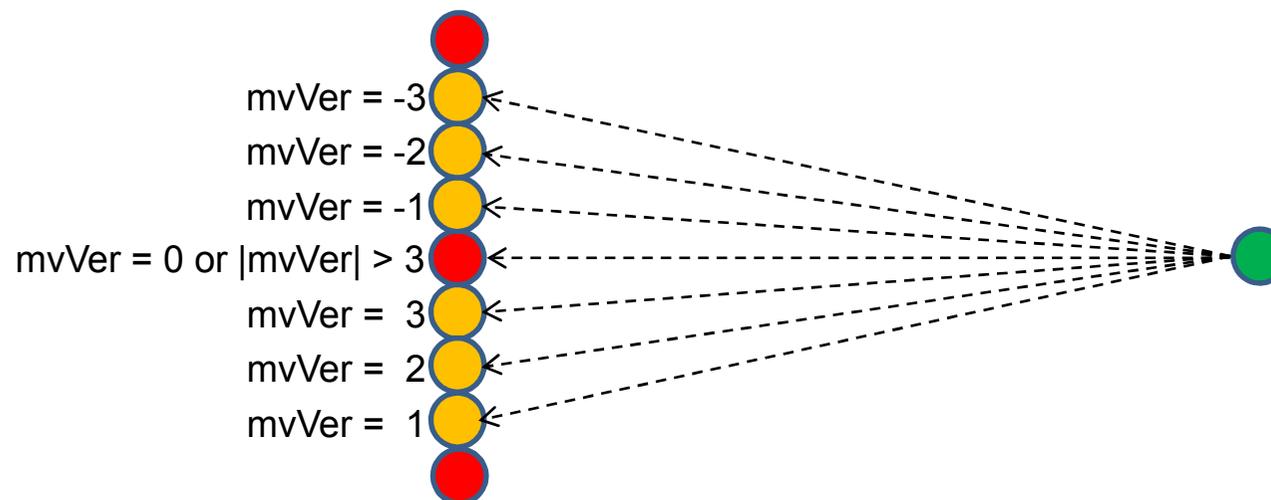
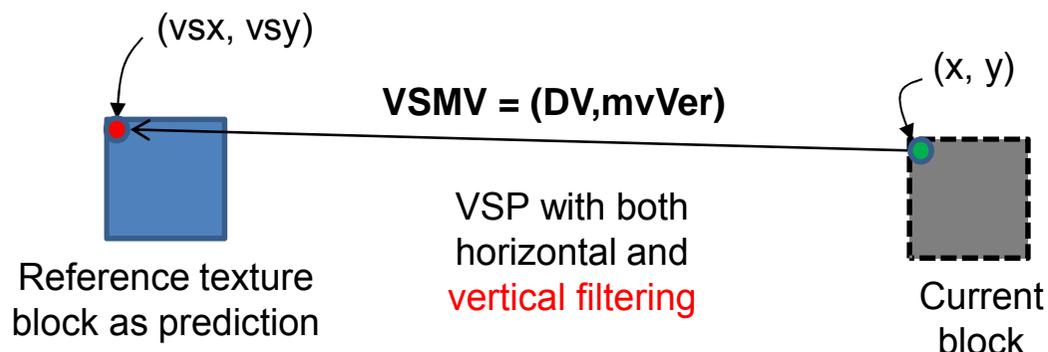
# Detailed procedure of the proposed VSP in luma

$NBDV = (mvHor, mvVer)$

$vsx = x + (dv \gg 2),$   
 $xFrac = dv \& 3;$

If  $(-3 \leq mvVer \leq 3),$   
 $vsy = y + (mvVer \gg 2),$   
 $yFrac = mvVer \& 3;$

Else  
 $vsy = y,$   
 $yFrac = 0.$



# Luma sample interpolation

- The samples labelled  $a_{0,0}$ ,  $b_{0,0}$ ,  $c_{0,0}$ ,  $d_{0,0}$ ,  $h_{0,0}$ , and  $n_{0,0}$  are derived by applying an 8-tap filter to the nearest integer position samples as follows:
  - $a_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 10 * A_{-1,0} + 58 * A_{0,0} + 17 * A_{1,0} - 5 * A_{2,0} + A_{3,0}) \gg \text{shift1}$
  - $b_{0,0} = (-A_{-3,0} + 4 * A_{-2,0} - 11 * A_{-1,0} + 40 * A_{0,0} + 40 * A_{1,0} - 11 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) \gg \text{shift1}$
  - $c_{0,0} = (A_{-2,0} - 5 * A_{-1,0} + 17 * A_{0,0} + 58 * A_{1,0} - 10 * A_{2,0} + 4 * A_{3,0} - A_{4,0}) \gg \text{shift1}$
  - $d_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 10 * A_{0,-1} + 58 * A_{0,0} + 17 * A_{0,1} - 5 * A_{0,2} + A_{0,3}) \gg \text{shift1}$
  - $h_{0,0} = (-A_{0,-3} + 4 * A_{0,-2} - 11 * A_{0,-1} + 40 * A_{0,0} + 40 * A_{0,1} - 11 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) \gg \text{shift1}$
  - $n_{0,0} = (A_{0,-2} - 5 * A_{0,-1} + 17 * A_{0,0} + 58 * A_{0,1} - 10 * A_{0,2} + 4 * A_{0,3} - A_{0,4}) \gg \text{shift1}$
- The samples labelled  $e_{0,0}$ ,  $i_{0,0}$ ,  $p_{0,0}$ ,  $f_{0,0}$ ,  $j_{0,0}$ ,  $q_{0,0}$ ,  $g_{0,0}$ ,  $k_{0,0}$ , and  $r_{0,0}$  are derived by applying an 8-tap filter to the samples  $a_{0,i}$ ,  $b_{0,i}$  and  $c_{0,i}$  with  $i = -3..4$  in the vertical direction as follows:
  - $e_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 10 * a_{0,-1} + 58 * a_{0,0} + 17 * a_{0,1} - 5 * a_{0,2} + a_{0,3}) \gg \text{shift2}$
  - $i_{0,0} = (-a_{0,-3} + 4 * a_{0,-2} - 11 * a_{0,-1} + 40 * a_{0,0} + 40 * a_{0,1} - 11 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) \gg \text{shift2}$
  - $p_{0,0} = (a_{0,-2} - 5 * a_{0,-1} + 17 * a_{0,0} + 58 * a_{0,1} - 10 * a_{0,2} + 4 * a_{0,3} - a_{0,4}) \gg \text{shift2}$
  - $f_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 10 * b_{0,-1} + 58 * b_{0,0} + 17 * b_{0,1} - 5 * b_{0,2} + b_{0,3}) \gg \text{shift2}$
  - $j_{0,0} = (-b_{0,-3} + 4 * b_{0,-2} - 11 * b_{0,-1} + 40 * b_{0,0} + 40 * b_{0,1} - 11 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) \gg \text{shift2}$
  - $q_{0,0} = (b_{0,-2} - 5 * b_{0,-1} + 17 * b_{0,0} + 58 * b_{0,1} - 10 * b_{0,2} + 4 * b_{0,3} - b_{0,4}) \gg \text{shift2}$
  - $g_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 10 * c_{0,-1} + 58 * c_{0,0} + 17 * c_{0,1} - 5 * c_{0,2} + c_{0,3}) \gg \text{shift2}$
  - $k_{0,0} = (-c_{0,-3} + 4 * c_{0,-2} - 11 * c_{0,-1} + 40 * c_{0,0} + 40 * c_{0,1} - 11 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) \gg \text{shift2}$
  - $r_{0,0} = (c_{0,-2} - 5 * c_{0,-1} + 17 * c_{0,0} + 58 * c_{0,1} - 10 * c_{0,2} + 4 * c_{0,3} - c_{0,4}) \gg \text{shift2}$

$A_{0,0}$	$a_{0,0}$	$b_{0,0}$	$c_{0,0}$
$d_{0,0}$	$e_{0,0}$	$f_{0,0}$	$g_{0,0}$
$h_{0,0}$	$i_{0,0}$	$j_{0,0}$	$k_{0,0}$
$n_{0,0}$	$p_{0,0}$	$q_{0,0}$	$r_{0,0}$

# Computations for each pixel

	Multipliers	Adders	Shifter
Horizontal Filter	8	7	1
Horizontal + Vertical Filter	64+8	56+7	8+1