

REDEFINING MOBILITY



# JCT3V-F0134: CE5 related: Wedgelet pattern extension from 4x4 block

Xin Zhao, Li Zhang and Ying Chen

# Introduction

- In DMM, different sets of wedgelet patterns are generated for different PU sizes
  - Each wedgelet pattern is derived by a pair of start and end point position
  - One wedgelet pattern list is maintained for each PU size

| PU size | Num of wedgelet patterns |
|---------|--------------------------|
| 4x4     | 86                       |
| 8x8     | 782                      |
| 16x16   | 1394                     |
| 32x32   | 1503                     |

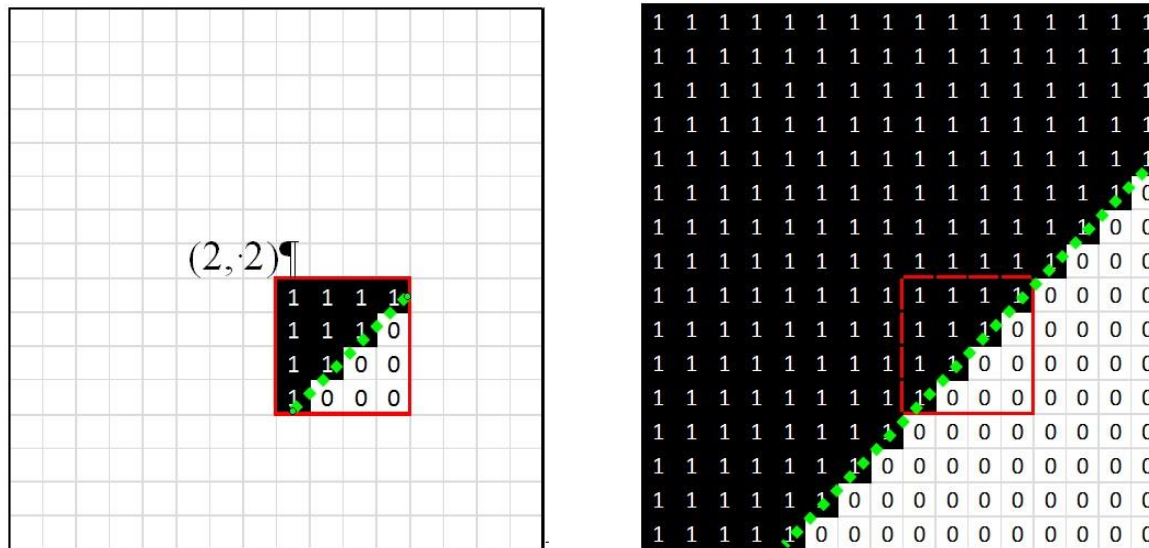
- During wedgelet pattern list construction, each generated wedgelet pattern is compared to all existing wedgelet pattern, and duplicate wedgelet pattern is excluded

# Problem statement

- For large block sizes, the wedgelet pattern list is considerably large, e.g., 1503 wedgelet patterns for 32x32 block size, which adds the storage cost
- The computation cost for generating a wedgelet pattern is somewhat large, especially for larger block sizes
- During the wedgelet pattern list generation process, each generated wedgelet pattern is compared to all the existing patterns (up to 1052) in the wedgelet list, which involves relatively large computational cost, especially for large block sizes

# Proposal

- Generate an NxN wedgelet pattern by extending a given 4x4 wedgelet pattern located at a given position
  - A 4x4 wedgelet pattern *subldx* and a position  $(i, j)$  is signaled
  - The 4x4 wedgelet pattern located at  $(i < 2, j < 2)$  is extended to NxN along the partition boundary line



- Applied to DMM1, SDC\_DMM1, DMM3

# Proposal

- Given a 4x4 wedgelet pattern, the partition boundary line function  $y=a*x+b$  is derived using the start point  $(xS, yS)$ , end point  $(xE, yE)$  and the 4x4 block position  $(i, j)$
- For each position in the NxN partition pattern, the partition is derived as  $y - a*x < b ? 1 : 0$ , where  $a$  and  $b$  is derived by

$$a = \frac{yS - yE}{xS - xE} \quad b = \frac{xS \cdot yE - yS \cdot xE}{xS - xE}$$

- Simplification to 4x4 wedgelet generation (also proposed in JCT3V-0133)
  - In HTM-8.0, a 4x4 wedgelet pattern  $p$  is derived as a downsampled 8x8 wedgelet pattern  $P$

$$p[x][y] = P[2 \cdot x + \text{offsetX}][2 \cdot y + \text{offsetY}],$$

where  $\text{offsetX}$ ,  $\text{offsetY}$  depends on wedgelet pattern  $P$ .

- The downsampling is proposed to be simplified as  $p[x][y] = P[2x][2y]$

# Results

- HTM-8.0, Common test condition

|                | video 1     | video 2     | video PSNR / video<br>bitrate | video PSNR / total<br>bitrate | synth PSNR / total<br>bitrate |
|----------------|-------------|-------------|-------------------------------|-------------------------------|-------------------------------|
| Balloons       | -0.1%       | 0.1%        | 0.0%                          | 0.0%                          | 0.0%                          |
| Kendo          | -0.1%       | -0.1%       | 0.0%                          | 0.0%                          | 0.0%                          |
| Newspapercc    | 0.0%        | 0.1%        | 0.0%                          | 0.0%                          | 0.2%                          |
| GhostTownFly   | 0.1%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| PoznanHall2    | 0.0%        | -0.2%       | 0.0%                          | -0.1%                         | -0.2%                         |
| PoznanStreet   | 0.1%        | 0.0%        | 0.0%                          | 0.0%                          | 0.0%                          |
| UndoDancer     | 0.3%        | 0.0%        | 0.0%                          | 0.1%                          | 0.2%                          |
| 1024x768       | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| 1920x1088      | 0.1%        | -0.1%       | 0.0%                          | 0.0%                          | 0.0%                          |
| <b>average</b> | <b>0.1%</b> | <b>0.0%</b> | <b>0.0%</b>                   | <b>0.0%</b>                   | <b>0.0%</b>                   |
| <i>Shark</i>   | <i>0.1%</i> | <i>0.2%</i> | <i>0.0%</i>                   | <i>0.1%</i>                   | <i>0.2%</i>                   |

# Results

- HTM-8.0, All-Intra test condition

|                | video 1     | video 2     | video PSNR / video<br>bitrate | video PSNR / total<br>bitrate | synth PSNR / total<br>bitrate |
|----------------|-------------|-------------|-------------------------------|-------------------------------|-------------------------------|
| Balloons       | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| Kendo          | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.2%                          |
| Newspapercc    | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| GhostTownFly   | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.2%                          |
| PoznanHall2    | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| PoznanStreet   | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| UndoDancer     | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.2%                          |
| 1024x768       | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| 1920x1088      | 0.0%        | 0.0%        | 0.0%                          | 0.0%                          | 0.1%                          |
| <b>average</b> | <b>0.0%</b> | <b>0.0%</b> | <b>0.0%</b>                   | <b>0.0%</b>                   | <b>0.1%</b>                   |
| <i>Shark</i>   | <i>0.0%</i> | <i>0.0%</i> | <i>0.0%</i>                   | <i>0.1%</i>                   | <i>0.3%</i>                   |

- Thanks HHI for crosscheck (JCT3V-F0223)

# Complexity Analysis

- Memory cost of storing wedgelet patterns
  - The total amount of storage used for wedgelet patterns in current 3D-HEVC is 243.42 Kbytes ( $= (4*4*86 + 8*8*782 + 16*16*1394 + 32*32*1503)/8$  bytes)
  - The total amount of storage used for wedgelet patterns in proposed method is 0.172 Kbytes, which is 99.93% memory reduction
- Computational cost for excluding duplicate wedgelet pattern
  - In 3D-HEVC, each generated NxN wedgelet pattern is compared to all the existing wedgelet patterns in the wedgelet list, and the computation load is relatively large
  - In the proposed method, this above part of computational cost is only needed for 4x4 wedgelet list generation



# Conclusion

- Proposed a simplified and generic wedgelet pattern generation method for all block sizes
- Complexity reduction
  - Memory cost for storing wedgelet patterns is largely reduced
  - Computational cost for excluding duplicate wedgelet pattern is largely reduced
- Coding performance
  - Almost no coding performance difference compared to HTM-8.0

# Thanks!