|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extension Development**  **of ITU-T SG 16 WP 3 and ISO/IECJTC 1/SC 29/WG 11**  3nd Meeting: Geneva, 17-23 Jan. 2013 | Document: JCT3V-C0044 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **CE6.h results on Simplification of Depth Modeling Mode 3** | | |
| *Status:* | Input Document | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Shiqi Wang ([sqwang@jdl.ac.cn](mailto:sqwang@jdl.ac.cn)) Hongbin Liu ([hongbin.liu@lge.com](mailto:hongbin.liu@lge.com))  Siwei Ma ([swma@jdl.ac.cn](mailto:swma@jdl.ac.cn))  Jie Jia (jie.jia@lge.com) |  |  |
| *Source:* | LG Electronics, Peking University | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

This contribution reports result of JCT3V-B0064. In the HTM-5.0.1, instead of performing exhaustive search from a huge number of candidate Wedgelet patterns, only candidate Wedgelet patterns that are within a predefined search range are searched. This contribution proposes to further simplify the Wedgelet pattern search at decoder by signalling the best pattern in the predefined search range. It is reported that proposed method has negligible influence on compression efficiency (on video, synthesized view, coded and synthesized view respectively) when compared with HTM-5.0.1.

# Introduction

The basic principle of DMM3 mode is to predict best Wedgelet pattern of depth block from co-located texture luma block (CTLB). In DMM3, the prediction unit is partitioned into two regions by a straight line and each region is predicted by a constant value, as shown in Fig. 1. The partition information, i.e., the “best” Wedgelet pattern of DMM3 is not signaled to the decoder. Instead, both encoder and decoder derive the “best” Wedgelet pattern by searching the candidate Wedgelet patterns on reconstructed CTLB. The Wedgelet pattern with the minimum cost is selected as the “best” pattern. In this way, no bit is required for indicating “best” Wedgelet pattern. However, the pattern search increases the decoder complexity.



Fig.1. Example of Wedgelet Pattern.

In the HTM-5.0.1, authors of [1] propose to only search candidate Wedgelet patterns that are within a predefined search range. In the case of intra coded CTLB (mode 2 to 34), the search range is defined by using the intra prediction mode of the top-left 4×4 block of the CTLB. If the top-left 4×4 block in CTLB is not intra-coded, the DMM3 mode is disabled. The search range is defined as follows. Assuming that the starting point of the Wedgelet pattern is *S*(*Xs, Ys*) and the ending point is *E*(*Xe, Ye*), then there is a pair of (*S, E*) corresponding to each Wedgelet pattern.

For each intra prediction mode (mode 2 to 34), the direction information (*Hi*, *Vi*) is defined as in Table 1. The fitness between the Wedgelet pattern (defined by (*Xs, Ys*) and *E*(*Xe, Ye*)) and the intra mode (defined by (*Hi*, *Vi*)) is defined as:

*D*[*i*] = | *Vi*×(*Xs*-*Xe*) - *Hi*×(*Ye*-*Ys*) | (1)

Then, each Wedgelet pattern can be mapped to the nearest intra prediction mode by searching minimum *D* among all intra prediction modes. Correspondingly, for each intra prediction mode *j*, there exists a Wedgelet pattern set *S*(*j*) in which every pattern’s nearest intra prediction mode is *j*. For intra prediction mode 0 and 1, *S*(0) and *S*(1) are defined to be contain only Wedgelet index 0, which corresponds to the pattern that has top-left sample in one partition and all the remaining pixels in the other partition.

In the HTM-5.0.1, only Wedgelet patterns in set *S*(*j*) are searched for DMM3, where *j* is the intra prediction mode of the top-left 4×4 block of the CTLB.

Table 1. *H,V* For each intra mode

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Intra Mode Index | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| *H* | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 |
| *V* | 32 | 26 | 21 | 17 | 13 | 9 | 5 | 2 | 0 | -2 | -5 | -9 | -13 | -17 | -21 | -26 | -32 |
| Intra Mode Index | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 |  |
| *H* | 26 | 21 | 17 | 13 | 9 | 5 | 2 | 0 | 2 | 5 | 9 | 13 | 17 | 21 | 26 | 32 |  |
| *V* | -32 | -32 | -32 | -32 | -32 | -32 | -32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 |  |

When the CTLB is not intra coded with mode 2~34, a coarse to fine search is performed. Firstly, a coarse pattern list is constructed and searched for the minimum distortion partition; secondly, the resulting Wedgelet partition is refined. The coarse patterns are generated every two start and end position, and the fine pattern consists up to eight patterns.

# Proposed Solution

In this contribution, we further simplify the DMM3 mode. The first solution only applies to the case when the CTLB is intra coded with mode 2~34. The second and third solutions are more general and covers all the cases of the CTLB coding modes. The motivation behind our approach lies in two points. Firstly, in DMM3, the “best” Wedgelet prediction pattern is searched on reconstructed CTLB rather than on original depth block. Therefore, the selected “best” pattern may not be the optimal pattern. Secondly, at the decoder, although [1] reduces the number of candidate Wedgelet patterns dramatically, it still has to perform Wedgelet pattern search. Based on above two considerations, we make two modifications based on [1] to further simplify the decoder of DMM3.

**Solution 1.**

This solution only applies to the case when the CTLB is intra coded with mode 2~34.

1. At the encoder, we search the best Wedgelet pattern on original depth block (instead of on reconstructed CTLB) as follows:

 (2)

where *ID*(*i*) represents the original depth value of pixel *i*, *ID*(*R1*) represents the DC value of the first partition region and *ID*(*R2*) represents the DC value of the second partition region.

1. Signal index of the best Wedgelet pattern in the predefined candidate Wedgelet pattern set. Therefore, the decoder does not perform any Wedgelet pattern search.

**Solution 2.**

When the CTLB is intra coded with mode 2~34, the coding scheme is the same with solution 1.

When the CTLB is not intra coded with mode 2~34, for the coarse search process:

1. At the encoder, we search the best Wedgelet pattern on original depth block (instead of on reconstructed CTLB) as follows:

 (2)

where *ID*(*i*) represents the original depth value of pixel *i*, *ID*(*R1*) represents the DC value of the first partition region and *ID*(*R2*) represents the DC value of the second partition region.

1. Signal index of the best Wedgelet pattern in the coarse candidate Wedgelet pattern set. Therefore, the decoder does not perform any Wedgelet pattern search at this stage.

For the fine search process, we apply the same method as in HTM-5.0.1.

**Solution 3.**

When the CTLB is intra coded with mode 2~34, the coding scheme is the same with solution 1.

When the CTLB is not intra coded with mode 2~34, for the coarse search, the method is same with solution 2. For the fine search, we disable it.

# Experimental Results

Proposed method is implemented into HTM 5.0.1 software, and proposed method is compared with HTM 5.0.1. The test condition is CTC [2] and All Intra.

## Performance of solution 1

Proposed method is compared with HTM-5.0.1 in Table 1 for CTC case and in Table 2 for all intra case. As can be seen, proposed method has negligible influence on decoding time and compression efficiency for CTC. This is because DMM3 is selected with small probability. For all intra case, proposed method can save 1% decoding time with negligible influence on compression efficiency.

Table 1: comparison of HTM-5.0.1 with proposed method (CTC)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 97.6% | 100.6% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 103.3% | 100.4% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.3% | 100.5% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.3% | 100.6% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 103.6% | 100.7% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.0% | 103.6% | 100.4% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.1% | 99.1% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.4% | 100.5% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.9% | 100.2% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **100.7%** | **100.3%** |

Table 2: comparison of HTM-5.0.1 with proposed method (all intra)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 97.5% | 99.3% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.5% | 99.8% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.9% | 98.6% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.1% | 99.1% | 98.0% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 102.1% | 99.4% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 102.6% | 98.6% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.3% | 98.5% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.6% | 99.3% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.8% | 98.6% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **100.3%** | **98.9%** |

## Performance of solution 2 (not crosschekced)

Table 3 and Table 4 show the performance for CTC case and all intra case. As can be seen, proposed method has negligible influence on decoding time and compression efficiency for CTC. For all intra case, proposed method can save 2% decoding time with negligible influence on compression efficiency.

Table 3: comparison of HTM-5.0.1 with proposed method (CTC)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.2% | 100.0% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 103.4% | 99.9% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.2% | 99.8% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.3% | 100.1% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.7% | 100.2% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.6% | 100.2% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.6% | 0.4% | 101.3% | 98.8% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.6% | 99.9% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.1% | 101.0% | 99.8% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.1%** | **0.1%** | **100.8%** | **99.9%** |

Table 4: comparison of HTM-5.0.1 with proposed method (all intra)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 102.5% | 98.8% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.0% | 99.2% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.3% | 97.4% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.1% | 101.1% | 96.8% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 102.2% | 99.5% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.1% | 97.4% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.2% | 97.2% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.6% | 98.4% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.0% | 100.9% | 97.7% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **100.8%** | **98.0%** |

## Performance of solution 3

Table 5 and Table 6 show the performance for CTC case and all intra case.

Table5: comparison of HTM-5.0.1 with proposed method (CTC)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 101.5% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | -0.1% | 0.0% |  | 101.7% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 101.7% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 101.6% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 102.1% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 101.1% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.6% | 0.4% |  | 101.0% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |  | 101.6% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.1% |  | 101.4% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.1%** | **0.0%** |  | **101.5%** |

Table 6: comparison of HTM-5.0.1 with proposed method (all intra)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video only | synthesized only | coded & synthesized | enc time | dec time |
| Balloons | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 97.5% | 98.9% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.1% | 98.8% |
| Newspapercc | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 103.4% | 97.4% |
| GhostTownFly | 0.0% | 0.0% | 0.0% | 0.0% | 0.1% | 0.1% | 98.2% | 97.0% |
| PoznanHall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.5% | 99.6% |
| PoznanStreet | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 101.0% | 97.3% |
| UndoDancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.6% | 97.1% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.6% | 98.3% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.8% | 97.7% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **99.7%** | **98.0%** |

# Conclusion

This proposal significantly reduces Wedgelet pattern search in DMM3 at decoder and it has negligible influence on compression efficiency. We recommend that proposed method is adopted into 3DV-HEVC.

# Reference

[1] P. Merkle, K. Müller, X. Zhao, Y. Chen, L. Zhang, M. Karczewicz, CE6.H results on simplified Wedgelet search for DMM modes 1 and 3, Doc. JCT3V-B0039, Shanghai, CN, 13–19 Oct. 2012.

[2] D. Rusanovskyy, K. Müller, A. Vetro, “Common Test Conditions of 3DV Core Experiments”, Doc. JCT3V-B1100, Shanghai, CN, 13–19 Oct. 2012.

# Patent rights declaration(s)

**LG Electronics / LG Electronics (China) R&D Center and Peking University may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**

# Annex: syntax specification

Coding quadtree syntax

|  |  |
| --- | --- |
| for( j = 0; j <= pbOffset; j = j + pbOffset ) |  |
| for( i = 0; i <= pbOffset; i = i + pbOffset ) |  |
| if ( dmm\_flag**[** x0 + i ][ y0+ j ] && (  dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WFULL ||   dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WFULLDELTA ) ) |  |
| **wedge\_full\_tab\_idx[**x0 + i **][**y0 + i **]** | ae(v) |
| for( j = 0; j <= pbOffset; j = j + pbOffset ) |  |
| for( i = 0; i <= pbOffset; i = i + pbOffset ) |  |
| if( dmm\_flag**[** x0 + i ][ y0+ j ] && (  dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WPREDTEXDELTA ||   dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WPREDTEX ) ) |  |
| If( texturePredMode**[** x0 + i ][ y0+ j ] == MODE\_INTRA && textureIntraPredMode**[** x0 + i ][ y0+ j ] >= 2 && textureIntraPredMode**[** x0 + i ][ y0+ j ] <= 34 ) |  |
| wedge\_predtex\_tab\_idx[ x0 + i ][ y0 + i ] | ae(v) |
| for( j = 0; j <= pbOffset; j = j + pbOffset ) |  |
| for( i = 0; i <= pbOffset; i = i + pbOffset ) |  |
| if( dmm\_flag**[** x0 + i ][ y0+ j ] && (  dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WPREDDIR ||   dmm\_mode**[** x0 + i ][ y0+ j ] = = MODE\_DMM\_WPREDDIRDELTA ) ) |  |
| **dmm\_delta\_end\_flag[**x0 + i **][**y0 + i **]** | ae(v) |

* + 1. Coding unit semantics

**wedge\_full\_tab\_idx**[ x0 ][ y0 ]specifies the index of the wedgelet pattern in the corresponding pattern list.

**wedge\_predtex\_tab\_idx**[ x0 ][ y0 ] specifies the index of the wedgelet pattern when dmm\_mode[x0][y0] is equal to 2 or 3, and texturePredMode [ x0 ][ y0 ] is equal to MODE\_INTRA and textureIntraPredMode [ x0 ][ y0 ] is in the range of 2 to 34.

The variable textureComp is set equal to the texture view component with view order index equal to ViewIdx and DepthFlag equal to 0. The variable texturePredMode is set equal to the PredMode[ xB ][ yB ] of textureComp. The variable textureIntraPredMode is set equal to IntraPredMode[ xB ][ yB ] of the textureComp.

* + - * 1. Specification of Intra\_DepthPartition (37, 38) prediction mode

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nT−1, are derived by the following ordered steps.

* 1. ~~Depending on texturePredMode and textureIntraPredMode, the variable dmmWedgeletPattern[ x ][ y ] specifying a binary partition pattern is derived as specified in the following.~~ 
     + ~~If texturePredMode is equal to MODE\_INTRA and textureIntraPredMode is in the range of 2 to 34, the following applies.~~ 
       - ~~The list candWedgeIndList specifying indices of candidate wedgelets and the variable numCandWedgeInd specifying the number of elements in candWedgeIndList are derived by~~

~~candWedgeIndList = TextModePredWedgeIndTable[ Log2( nT ) ][ texturePredMode ] (G‑46)  
numCandWedgeInd = NumTextModePredWedgeInd[ Log2( nT ) ][ texturePredMode ] (G‑47)~~

* + - ~~Otherwise ( texturePredMode is not equal to MODE\_INTRA or textureIntraPredMode is not in the range of 2 to 34), the following applies.~~
      * ~~The list coarseCandWedgeIndList specifying indices of candidate wedgelets and the variable numCoarseCandWedgeInd specifying the number of elements in coarseCandWedgeIndList are derived by~~

~~coarseCandWedgeIndList = CoarseWedgeIndTable[ Log2( nT ) ] (G‑48)  
numCoarseCandWedgeInd = NumCoarseWedgeInd[ Log2( nT ) ] (G‑49)~~

* + - * ~~The derivation process for a texture predicted wedgelet index as specified in subclause is invoked with the luma location ( xB, yB ), the transform size nT, the candidate wedgelet indices coarseCandWedgeList and the number of candidate wedgelet indices numCoarseCandWedgeInd as inputs and the output is assigned to coarseWedgeIdx.~~
      * ~~The list candWedgeIndList specifying indices of candidate wedgelets and the variable numCandWedgeInd specifying the number of elements in candWedgeIndList are derived by~~

~~candWedgeIndList = RefinedWedgeIndTable[ Log2( nT ) ][ coarseWedgeIdx ] (G‑50)  
numCandWedgeInd = NumRefinedWedgeInd[ Log2( nT ) ][ coarseWedgeIdx ] (G‑51)~~

* 1. ~~The derivation process for a texture predicted wedgelet index as specified in subclause XXX is invoked with the luma location ( xB, yB ), the transform size nT, the candidate wedgelet indices candWedgeIndList and the number of candidate wedgelet indices numCandWedgeInd as inputs and the output is assigned to wedgeIdx.~~
  2. ~~The variable wedgePattern [ x ][ y ] with x, y =0..nT−1, specifying a binary partition pattern is derived as.~~
     + 1. ~~wedgePattern = WedgePatternTable[ Log2( nT) ][ wedgeIdx]~~
  3. ~~The depth partition value derivation and assignment process as specified in subclause  is invoked with the neighbouring samples p[ x ][ y ], the binary pattern wedgePattern[ x ][ y ], the transform size nT, the dcOffsetAvailFlag set equal to ( intraPredMode = = Intra\_DepthPartition(38) ), and the DC Offsets QuantDcOffsetP0[ xB ][ yB ], and QuantDcOffsetP1[ xB ][ yB ] as inputs and the output is assigned to predSamples[ x ][ y ].~~
  4. Depending on texturePredMode and textureIntraPredMode, the variable dmmWedgeletPattern[ x ][ y ] specifying a binary partition pattern is derived as specified in the following.
     + If texturePredMode is equal to MODE\_INTRA and textureIntraPredMode is in the range of 2 to 34, the following applies.
       - The list candWedgeIndList specifying indices of candidatewedgelets and the variable numCandWedgeInd specifying the number of elements in candWedgeIndList are derived by

candWedgeIndList = TextModePredWedgeIndTable[ Log2( nT ) ][ texturePredMode ] (G‑46)  
numCandWedgeInd = NumTextModePredWedgeInd[ Log2( nT ) ][ texturePredMode ] (G‑47)

wedgeIdx = candWedgeIndList[ wedge\_predtex\_tab\_idx[ xB ][ yB ] ] (G-48)

* + - Otherwise ( texturePredMode is not equal to MODE\_INTRA or textureIntraPredMode is not in the range of 2 to 34), the following applies.
      * The list coarseCandWedgeIndList specifying indices of candidatewedgelets and the variable numCoarseCandWedgeInd specifying the number of elements in coarseCandWedgeIndList are derived by

coarseCandWedgeIndList = CoarseWedgeIndTable[ Log2( nT ) ] (G‑49)  
numCoarseCandWedgeInd = NumCoarseWedgeInd[ Log2( nT ) ] (G‑50)

* + - * The derivation process for a texture predicted wedgelet index as specified in subclause G.8.4.4.2.8.1is invoked with the luma location ( xB, yB ), the transform size nT, the candidate wedgelet indices coarseCandWedgeList and the number of candidate wedgelet indices numCoarseCandWedgeInd as inputs and the output is assigned to coarseWedgeIdx.
      * The list candWedgeIndList specifying indices of candidatewedgelets and the variable numCandWedgeInd specifying the number of elements in candWedgeIndList are derived by

candWedgeIndList = RefinedWedgeIndTable[ Log2( nT ) ][ coarseWedgeIdx ] (G‑51)  
 numCandWedgeInd = NumRefinedWedgeInd[ Log2( nT ) ][ coarseWedgeIdx ] (G‑52)

* + - * The derivation process for a texture predicted wedgelet index as specified in subclause XXX is invoked with the luma location ( xB, yB ), the transform size nT, the candidate wedgelet indices candWedgeIndList and the number of candidate wedgelet indices numCandWedgeInd as inputs and the output is assigned to wedgeIdx.
  1. The variable wedgePattern[ x ][ y ] with x, y =0..nT−1, specifying a binary partition pattern is derived as.
     + 1. wedgePattern= WedgePatternTable[ Log2( nT) ][ wedgeIdx]
  2. The depth partition value derivation and assignment process as specified in subclause G.8.4.4.2.8.1 is invoked with the neighbouring samples p[ x ][ y ], the binary pattern wedgePattern[ x ][ y ], the transform size nT, the dcOffsetAvailFlag set equal to ( intraPredMode = = Intra\_DepthPartition(38) ), and the DC Offsets QuantDcOffsetP0[ xB ][ yB ], and QuantDcOffsetP1[ xB ][ yB ]as inputs and the output is assigned to predSamples[ x ][ y ].