|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extensions**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  6th Meeting: Geneva, CH, 25 Oct. – 1 Nov. 2013 | Document: JCT3V-F0138 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **AHG7: Updating Mechanism for Coding of Depth Lookup Table (Delta-DLT)** | | |
| *Status:* | Input Document | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Fabian Jäger Institut für Nachrichtentechnik RWTH Aachen University | Tel: Email: | +49 (0) 241 80 27678 jaeger@ient.rwth-aachen.de |
| *Source:* | RWTH Aachen University | | |

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

# Abstract

This contribution proposes a very simple updating mechanism for coding of the Depth Lookup Table (DLT). In B0036 the DLT was proposed as an efficient coding approach for sparse depth maps. In the current 3D-HEVC design, the DLT is coded in the SPS/VPS for each view independently. In this contribution it is proposed to explicitly code the DLT for the base view only and allow updating the DLT entries for the dependent views. By the proposed simple updating mechanism it is possible to reduce the coding overhead for the DLT in the VPS by more than 60%, depending on encoder decisions.

# Introduction

In 3D video coding, a depth map for each view needs to be encoded besides the conventional video data. These depth maps show different signal characteristics compared to video data as they contain piecewise smooth regions bounded by strong edges. As depth maps are often estimated from texture data or are pre-processed, their histogram might be relatively sparse. As a result, a Depth Lookup Table (DLT) was proposed [1] to exploit the histogram characteristics by only signaling difference indexes of the DLT instead of signaling the residual depth values themselves. By this approach the bit depth of these residual values can be reduced, which consequently results in higher coding efficiency.

The DLT is constructed at the encoder by analyzing the histogram of the original, uncompressed depth map. This DLT is afterwards transmitted to the decoder to allow for the mapping of indexes to actual depth values. In a multiview coding scenario, multiple depth maps may have similar, but still different depth map histograms and in these cases such an updating mechanism is beneficial to the overall coding performance.

# DLT Coding in 3D-HEVC

In the latest specification of the 3D extension for High Efficiency Video Coding, the DLT is coded in the VPS, independently for all views. In a multiview coding scenario the depth map characteristics of different views are mostly very similar to each other. Consequently, coding the DLT for each view independently introduces unnecessary coding overhead to the VPS.

# Description of DLT Update Mechanism

The proposed solution for updating DLT values between views, aims at reducing the required bitrate for the DLT.

The proposal uses a very simple approach for updating the DLT between views by signaling only the differences between the lookup tables.

Macintosh HD:Users:jaeger:Desktop:Unbenannt1.pdf

Figure 1: Delta-DLT coding mechanism

As depicted in Figure 1, DLT0 and DLT1 are DLTs of the base view and of the dependent view, respectively. They are both represented in a full bit-map (256 bit) form where a ‘1’ indicates that the corresponding value is present in the DLT and a ‘0’ indicates that the corresponding value is not present in the DLT.

A delta-DLT DLT1’ is calculated as DLT1’ = DLT0^DLT1, where ‘^’ denotes bit-wise “exclusive or”, as described in the HEVC specification text. Afterwards, DLT1’ instead of DLT1 is coded in the VPS. After DLT1’ is decoded, DLT1 is reconstructed as DLT1 = DLT0^DLT1’. Since DLT0 and DLT1 are often quite similar, the “exclusive or” operation results in a bit-map with many zeros, which is itself efficiently coded by the DLT coding method.

# Experimental Results

As previous simulations already showed that coding of the DLT does not have a measurable impact on the overall coding efficiency when following the common test conditions, this contribution only focuses on the coding efficiency of the DLT itself.  
For this purpose, additional counters have been integrated into HTM 6.0 (DLT coding has not changed between 6.0 and 8.0) to count the number of bits that are being spent for DLT coding.

## Configuration A: Conventional DLT Coding

In this configuration the initial implementation of DLT coding is used for each view. Other proposals already demonstrated that this is not the optimal coding method for DLT entries, but the proposed Delta-DLT approach is compatible with this method, too.

The third column shows the performance of the Delta-DLT method when the encoder decides to code all views’ DLT values in the base view’s DLT. In these cases there is no delta to be coded for the dependent views.

|  |  |  |
| --- | --- | --- |
| **Conventional DLT Coding** | | |
| **Sequence** | **Delta-DLT** | **Delta-DLT (Base)** |
| Balloons | 39,55% | 35,96% |
| Kendo | 46,13% | 36,76% |
| Newspaper | 37,49% | 36,30% |
| Poznan\_Hall2 | 43,27% | 35,73% |
| **AVERAGE** | **41,61%** | **36,19%** |

## Configuration B: Mediatek’s RCBM Coding

The second configuration uses RCBM coding for DLT values, as proposed by Mediatek in previous meetings.

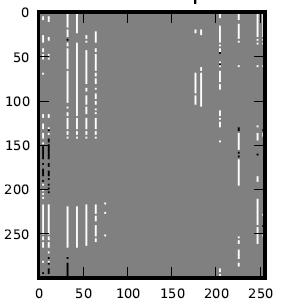
The third column again shows the performance of the Delta-DLT method when the encoder decides to code all views’ DLT values in the base view’s DLT.

The fourth and fifth column show an extension to the very simple Delta-DLT method, which allows to signal two unconnected Delta-DLTs for cases where the update values are distributed non-uniformly. This approach was already presented in document JCT3V-D0054 and it is to be emphasized that both, Delta-DLT and Multi-Range Delta-DLT, are independent of the underlying single-view DLT coding method.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sequence** | **Delta-DLT** | **Delta-DLT (Base)** | **Multi-Range Delta-DLT** | **Multi-Range Delta-DLT (Base)** |
| Balloons | 46,48% | 35,72% | 46,76% | 35,72% |
| Kendo | 100,28% | 36,84% | 52,35% | 36,84% |
| Newspaper | 40,40% | 35,89% | 40,73% | 35,89% |
| Poznan\_Hall2 | 77,53% | 35,76% | 54,43% | 35,76% |
| **AVERAGE** | **66,17%** | **36,05%** | **48,57%** | **36,05%** |

# Multi-range Delta-DLT Coding

The original range-based approach (binary string) of RCBM [2] used to represent the values that are signaled in the inter-view DLT updates in the non-base views is modified. In the proposed method, the original single range representation can be divided into 2 sub-ranges in order to exploit the characteristics of the inter-view DLT updates (see Fig.1). The decision whether to use single or multi-range representation is made based on the number of bits required to represent the DLT using each representation. In this method, for each DLT update in the non-base view, first sub-range is signaled in the same way as the original single-range of RCBM. Second sub-range is signaled by indicating the offset to the previous sub-range and the width of the current sub-range (width is calculated based on the difference between minimum and maximum value in the sub-range). The values in the sub-range are signaled using the representation as in the binary string of RCBM. Consequently, the original range is divided into 2 sub-ranges, where the division point between the sub-ranges is determined by the largest gap between the signaled values in DLT (in this sense gap between signaled DLT values means a number of neighboring depth values that are not signaled in DLT). Fig. 2b shows an example of the proposed approach using 2 sub-ranges.



Time

Depth level

Figure 2: Examplary inter-view DLT update (Kendo, view 5).

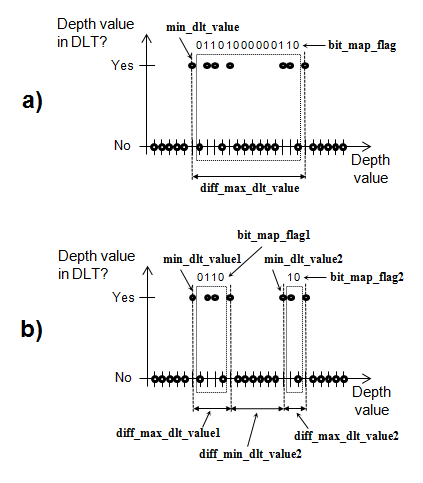
****

Figure 3: DLT coding: a) single-range, b) multi-range (with 2 sub-ranges).

# Cross Check

The cross check of the proposed Delta-DLT coding scheme was performed by Huawei Technologies. They investigated the source code modifications for the bit counting and ran the simulations for verification of the presented results.

In their investigation they did not find any problems with the source code. Their simulation results perfectly match with those presented in this document.

# Working Draft Changes

**H.8.3.6 Decoding process for a depth lookup table**

* ~~For i = 0..num\_depth\_values\_in\_dlt –1 the elements in Idx2DepthValue are derived as follows.~~
  + ~~Idx2DepthValue[ i ] is set equal to dlt\_depth\_value[ i ]~~
* The elements in Idx2DepthValue are derived as follows.
* If nuh\_layer\_id >> 1 is not equal to 0
  + for i =0… MAX\_DEPTH\_VALUE-1
    - BitMapFlag[i] = BitMapFlag[i] ^ BitMapFlag’[i], where BitMapFlag’ represents the BitMapFlag in the sequence with nuh\_layer\_id equal to 1.
* Set Idx= 0;
* for i =0… MAX\_DEPTH\_VALUE-1
  + If BitMapFlag[i]==1, then Idx2DepthValue[Idx] = i and Idx++;
* num\_depth\_vaules\_in\_dlt=Idx;

# Conclusion

In this proposal, an efficient inter-view coding method for the depth lookup table (DLT) is proposed. The proposed method can reduce the required bitrate for DLT syntax elements by up to about 60% compared to independent coding of DLTs for each view in a 3-view coding configuration.

**References**

[1] F. Jäger, “3D-CE6.h Results on Simplified Depth Coding with an optional Depth Lookup Table,” Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) of ITU-T VCEG and ISO/IEC MPEG, Shanghai, China, JCT3V-B0036, 2012.

[2] K. Zhang, J. An, and S. Lei, “3D-CE6.h related: An efficient coding method for DLT in 3DVC,” Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) of ITU-T VCEG and ISO/IEC MPEG, Geneva, Switzerland, JCT3V-C0142, 2013.