The proposed working draft modifications are as follows.

H.7.3.2.1.2 Video parameter set extension 2 syntax

|  |  |
| --- | --- |
| vps\_extension2( ) { | Descriptor |
| while( !byte\_aligned( ) ) |  |
| **vps\_extension\_byte\_alignment\_reserved\_one\_bit** | u(1) |
| for( i = 0; i <= vps\_max\_layers\_minus1; i++ ) { |  |
| … |  |
| if ( DepthLayerFlag[ layerId ] ) { |  |
| **vps\_dmm\_flag**[ layerId ] | u(1) |
| **lim\_qt\_pred\_flag**[ layerId ] | u(1) |
| **dlt\_flag**[ layerId ] | u(1) |
| if( dlt\_flag[ layerId ] ) { |  |
| **~~num\_depth\_values\_in\_dlt~~**~~[ layerId ]~~ | ~~ue(v)~~ |
| ~~for ( j = 0; j < num\_depth\_values\_in\_dlt[ layerId ] ; j++) {~~ |  |
| **~~dlt\_depth\_value~~**~~[ layerId ][ j ]~~ | ~~ue(v)~~ |
| **num\_dlt**[ layerId ] | u (4) |
| for ( j = 0; j < num\_dlt[ layerId ] ; j++) { |  |
| depth\_lookup\_talbe**(** layerId, j ) |  |
| } |  |
| } |  |
| } |  |
| } |  |
| **iv\_mv\_scaling\_flag** | u(1) |
| } |  |

H.7.3.5.1.1 Slice header extension syntax

|  |  |
| --- | --- |
| slice\_header\_extension( ) { | Descriptor |
| **…** |  |
| if( DepthLayerFlag[ layerId ] && dlt\_flag[ layerId ] ){ |  |
| **use\_sps\_dlt\_flag** | u(1) |
| if(use\_sps\_dlt\_flag && num\_dlt[layer\_id] > 1) |  |
| **sps\_dlt\_idx** | ue(v) |
| if( **!** use\_sps\_dlt\_flag ){ |  |
| if( slice\_type = = I ) |  |
| depth\_lookup\_talbe**(** layerId, num\_dlt[layer\_id] ) |  |
| else{ |  |
| **use\_ref\_pic\_dlt\_flag** | u(1) |
| if(!use\_ref\_pic\_dlt\_flag) |  |
| depth\_lookup\_talbe**(** layerId, num\_dlt[layer\_id] ) |  |
| else{ |  |
| if( slice\_type = = B ) |  |
| **dlt\_from\_l0\_flag** | u(1) |
| if( ( dlt\_inherit\_from\_l0\_flag && num\_ref\_idx\_l0\_active\_minus1 > 0 )  | | ( ! dlt\_inherit\_from\_l0\_flag && num\_ref\_idx\_l1\_active\_minus1 > 0 ) ) |  |
| **dlt\_ref\_idx** | ue(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |

H.7.3.5.5 Depth lookup table syntax

|  |  |
| --- | --- |
| depth\_lookup\_talbe**(** layerId, dlt\_id ) { | Descriptor |
| **code\_full\_bit\_map\_flag**[ layerId ][ dlt\_id ] | u(1) |
| if(!**code\_full\_bit\_map\_flag**[ layerId ][ dlt\_id ]){ |  |
| **min\_dlt\_value**[ layerId ][ dlt\_id ] | u(v) |
| **diff\_max\_dlt\_value**[ layerId ][ dlt\_id ] | u(v) |
| } |  |
| for(j=0;j< MaxDltValue[ layerId ] [ dlt\_id ] - min\_dlt\_value[ layerId  ][ dlt\_id ] – 1;j++) |  |
| **bit\_map\_flag**[ layered ][ dlt\_id ] [ j ] | u(1) |
| } |  |

**H.7.4.2.1.2 Video parameter set extension 2 semantics**

**~~num\_depth\_values\_in\_dlt~~**~~[ layerId ] specifies the number of different depth values and the number of elements in the depth lookup table for depth view components of the current layer with layer\_id equal to layerId.~~

**~~dlt\_depth\_value[~~**~~layerId~~**~~]~~**~~[ j ] specifies the j-th entry in the depth lookup table for depth view components with layer\_id equal to layerId.~~

**num\_dlt**[ layerId ] specifies the number of depth lookup tables coded in VPS. When num\_dlt[layered] is not present, it shall be inferred to be 0.

H.7.4.5.1.1 Slice header extension semantics

use\_sps\_dlt\_flag specifies whether to use a depth lookup table coded in SPS. When use\_sps\_dlt\_flag is not present, it shall be inferred to be 0.

**sps\_dlt\_idx** specifies the index of the depth lookup table to be used for the current slice. When sps\_dlt\_idx is not present, it shall be inferred to be 0. When use\_sps\_dlt\_flag is equal to 1, NumDltValue is set equal to NumDltValueSet[sps\_dlt\_idx], and Idx2DepthValue[i] is set equal to Idx2DepthValueSet[sps\_dlt\_idx][i], with i = 0,…, NumDltValue.

use\_ref\_pic\_dlt\_flag specifies whether to use a depth lookup table from a reference picture. When use\_ref\_pic\_dlt\_flag is not present, it shall be inferred to be 0.

**dlt\_from\_l0\_flag** equal to 1 specifies that the depth lookup talbe is inherited from list 0, otherwise it is inherited from list 1.When dlt\_from\_l0\_flag is not present, it is inferred to be 0.

**dlt\_ ref\_idx** specifies the reference index of the reference picture from which the depth lookup table is inherited. When the current slice is a P slice, dlt\_ref\_idx refers to a picture in list 0. When the current slice is a B slice, dlt\_ref\_idx refers to a picture in list 0 if dlt\_from\_l0\_flag is 1, otherwise it refers to a picture in list 1. dlt\_ref\_idx shall always refer to a valid list entry, and the resulting picture shall be the same for all slices of a coded picture. When dlt\_ref\_idx is not present, it is inferred to be 0.

H.7.4.5.5 Depth lookup table semantics

**code\_full\_bit\_map\_flag**[layerId][dlt\_id] specifies whether to code the full bit map or not in the depth lookup table referred by dlt\_id for depth view components with layer\_id equal to layerId.

**min\_dlt\_value**[layerId][dlt\_id] specifies the smallest value in the depth lookup table referred by dlt\_id for depth view components with layer\_id equal to layerId. The number of bits used to represent it is log2(MAX\_DEPTH\_VALUE + 1). When min\_dlt\_value[layerId][dlt\_id] is not present, it shall be inferred to be -1.

**diff\_max\_dlt\_value**[layerId][dlt\_id] specifies the difference between the largest and smallest value in the depth lookup table referred by dlt\_id for depth view components with layer\_id equal to layerId. MaxDltValue[layerId][dlt\_id] is computed as follows: MaxDltValue[layerId][dlt\_id] = min\_dlt\_value[layerId][dlt\_id] + diff\_max\_dlt\_value[layerId][dlt\_id]. The number of bits used to represent it is log2(MAX\_DEPTH\_VALUE+1-min\_dlt\_value[layerId][dlt\_id] ). When diff\_max\_dlt\_value[layerId][dlt\_id] is not present, MaxDltValue[layerId][dlt\_id] shall be inferred to be MAX\_DEPTH\_VALUE+1.

**bit\_map\_flag**[layerId][dlt\_id][j]specifies the j-th entry in the bit map in the depth lookup table referred by dlt\_id for depth view components with layer\_id equal to layerId.

**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 Idx2DepthValueSet[nuh\_layer\_id][k] are derived as follows.
  + Set Idx= 0;
  + If min\_dlt\_value[nuh\_layer\_id][k] >=0, then Idx2DepthValue[nuh\_layer\_id][k][0] is set equal to min\_dlt\_value and Idx++;
  + If MaxDltValue[nuh\_layer\_id][k] - min\_dlt\_value[nuh\_layer\_id][k] > 1, then for i = 0… MaxDltValue[nuh\_layer\_id][k] - min\_dlt\_value[nuh\_layer\_id][k] – 2
    - * If bit\_map\_flag[nuh\_layer\_id][k][i]==1, then Idx2DepthValue[nuh\_layer\_id] [k][Idx] = i + min\_dlt\_value[nuh\_layer\_id][k] + 1 and Idx++;
  + If MaxDltValue[nuh\_layer\_id][k] <= MAX\_DEPTH\_VALUE[nuh\_layer\_id][k] and MaxDltValue[nuh\_layer\_id][k] != min\_dlt\_value[nuh\_layer\_id][k], then Idx2DepthValueSet[nuh\_layer\_id][k][Idx] = MaxDltValue[nuh\_layer\_id][k] and Idx++;
  + NumDltValueSet[nuh\_layer\_id][k] = Idx;

If use\_sps\_dlt\_flag is equal to 1, NumDltValue is set equal to NumDltValueSet[nuh\_layer\_id][dlt\_idx], and Idx2DepthValue[ i ] is set equal to Idx2DepthValueSet[nuh\_layer\_id] [dlt\_idx][ i ], with i = 0,…, NumDltValue.

Else, if use\_ref\_pic\_dlt\_flag is equal to 0 or slice\_type is equal to I, NumDltValue is set equal to NumDltValueSet[nuh\_layer\_id] [num\_dlt[nuh\_layer\_id]], and Idx2DepthValue[ i ] is set equal to Idx2DepthValueSet[nuh\_layer\_id] [num\_dlt[ nuh\_layer\_id] ][ i ], with i = 0,…, NumDltValue.

Else, NumDltValue is set equal to that in RefPicListX[dlt\_ref\_idx ], with X equal to 1-dlt\_from\_l0\_flag. Idx2DepthValue[ i ] is set equal to Idx2DepthValue[ i ] in RefPicListX[dlt\_ref\_idx ], with X equal to 1-dlt\_from\_l0\_flag.

num\_depth\_values\_in\_dlt[ nuh\_layer\_id ] is replaced by NumDltValue in the following description.