

# Adaptive Clipping

## JVET-C0040

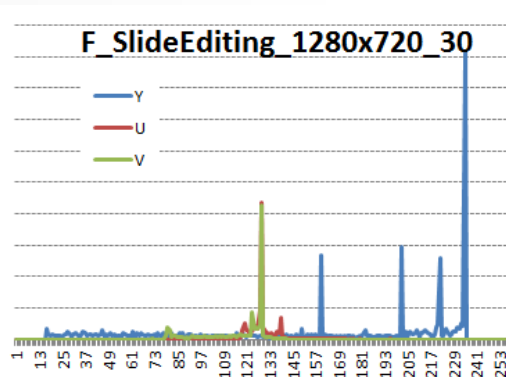
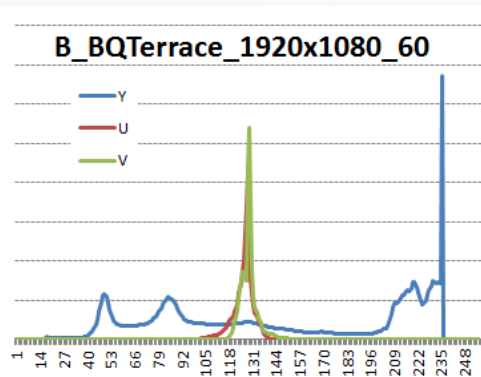
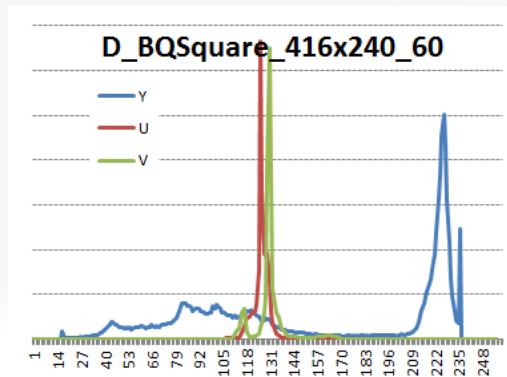


technicolor



## ■ Rationale

- Many video sequences have particular sample range limits characteristics
  - Camera capture settings vs scene dynamic range
  - Post-processing, grading and mastering treatments
  - Standardized signal limited range values (ex: BT.709...), range reserved values
  - ...



## ■ Current

- Clipping bounds depend on internal bit-depth

$$T_{clip} = Clip_{BD}( T, bitdepth ) = Clip3( 0, (1 \ll bitdepth) - 1, T )$$

## ■ Proposal

- Clipping bounds depend on actual original pictures bounds ( $min_C$ ,  $max_C$ ) of component  $C$
- Clipping bounds coded in the slice header

$$T_{clip} = Clip_{BD}( T, bitdepth, C ) = Clip3( min_C, max_C, T )$$

## ■ Unify the code with *ClipBD()* function

```
template <typename T> T ClipBD(const T x, const Int bitDepth, const ComponentID compID)
{
    if ( g_TchClipParam.isActive ) {
        switch(compID) {
            case COMPONENT_Y:  return Clip3( (T)g_TchClipParam.Y().m, (T)g_TchClipParam.Y().M, x ); break;
            case COMPONENT_Cb: return Clip3( (T)g_TchClipParam.U().m, (T)g_TchClipParam.U().M, x ); break;
            case COMPONENT_Cr: return Clip3( (T)g_TchClipParam.V().m, (T)g_TchClipParam.V().M, x ); break;
            default: assert(false);
        }
    }
    // default
    return Clip3((T)0,(T)((1 << bitDepth) - 1), x);
}
```

## Changes in the Code: unify the code with *ClipBD()* function

### ■ Current (*TComInterpolationFilter*)

```
Pel maxVal = (1 << bitDepth) - 1;
```

```
Pel minVal = 0;
```

```
if (val < minVal)
```

```
{
```

```
    val = minVal;
```

```
}
```

```
if (val > maxVal)
```

```
{
```

```
    val = maxVal;
```

```
}
```

### ■ Proposal

```
Val = ClipBD( val, bitDepth, compID );
```

## Changes in the Code: unify the code with *ClipBD()* function

### ■ Current (*TComAdaptiveLoopFilter*)

```
Val = (Pel) Clip3(0, m_nIBDMax, val);
```

### ■ Proposal

```
Val = ClipBD( val, bitDepth, compID );
```

## Changes in the Code: unify the code with *ClipBD()* function

### ■ Current (*TComLoopFilter*)

```
Val = ClipBD( val, bitDepth);
```

### ■ Proposal

```
Val = ClipBD( val, bitDepth, compID );
```

# Performances of Adapting Clipping

	All Intra Main10				
	Y	U	V	EncT	DecT
Class A1	-0.16%	-0.14%	-0.13%	105%	97%
Class A2	-0.05%	-0.03%	-0.04%	104%	98%
Class B	-0.18%	-0.06%	-0.06%	104%	98%
Class C	-0.12%	-0.01%	0.05%	103%	98%
Class D	-0.32%	-0.06%	-0.21%	100%	81%
Class E	-0.03%	0.02%	0.19%	101%	99%
<b>Overall</b>	-0.15%	-0.05%	-0.04%	103%	95%
Class F (optional)	-1.02%	-0.30%	-0.31%	103%	100%

	Random Access Main10				
	Y	U	V	EncT	DecT
Class A1					
Class A2	-0.29%	-0.22%	-0.12%	100%	99%
Class B	-0.30%	0.06%	-0.07%	101%	93%
Class C	-0.13%	-0.03%	0.08%	102%	96%
Class D	-0.73%	-0.58%	-0.63%	101%	86%
Class E					
<b>Overall (Ref)</b>					
Class F (optional)	-0.79%	-0.33%	-0.68%	103%	93%

	Low Delay B Main10				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	-0.26%	-0.07%	-0.01%	104%	93%
Class C	-0.15%	-0.07%	-0.31%	100%	93%
Class D	-0.50%	-0.33%	-0.35%	102%	93%
Class E	0.18%	0.45%	0.17%	102%	93%
<b>Overall (Ref)</b>	-0.21%	-0.04%	-0.14%	102%	93%
Class F (optional)	-0.76%	-0.32%	-0.94%	106%	95%



# Encoder improvement

## ■ Residuals smoothing

$$\text{Orig}_0 = \begin{bmatrix} 1000 & 700 & 500 \\ 800 & 1000 & 700 \\ 600 & 500 & 500 \end{bmatrix}$$

$$P_0 = \begin{bmatrix} 980 & 400 & 400 \\ 300 & 1020 & 400 \\ 400 & 400 & 400 \end{bmatrix}$$

$$\text{Res}_0 = \begin{bmatrix} 20 & 300 & 100 \\ 500 & -20 & 300 \\ 200 & 100 & 100 \end{bmatrix}$$

$$\text{TQ}_0 = \begin{bmatrix} 4 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & -2 \end{bmatrix} \quad \text{Dist}_0 = 5277$$

- It is counter productive to encode residuals that will be clipped
- Replace these residuals samples at clipping bounds with low-pass filtered values (average of other residuals values):

$$(300 + 100 + 500 + 300 + 200 + 100 + 100)/7 = 229$$

$$\text{Res}_1 = \begin{bmatrix} 229 & 300 & 100 \\ 500 & 229 & 300 \\ 200 & 100 & 100 \end{bmatrix}$$

$$\text{TQ}_1 = \begin{bmatrix} 5 & 1 & 1 \\ 1 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad \text{Dist}_1 = 4852$$

# Performances of Adapting Clipping with Residuals Smoothing

	All Intra Main10				
	Y	U	V	EncT	DecT
Class A1	-0.18%	-0.14%	-0.16%	108%	97%
Class A2	-0.05%	-0.05%	-0.11%	104%	98%
Class B	-0.20%	-0.11%	-0.02%	104%	98%
Class C	-0.15%	0.04%	-0.02%	105%	98%
Class D	-0.34%	-0.05%	-0.21%	105%	81%
Class E	-0.02%	-0.05%	0.15%	101%	99%
<b>Overall</b>	-0.17%	-0.06%	-0.07%	104%	95%
Class F (optional)	-0.86%	0.09%	0.00%	112%	100%

	Random Access Main10				
	Y	U	V	EncT	DecT
Class A1	-0.22%	-0.02%	-0.10%	104%	93%
Class A2	-0.26%	-0.28%	-0.25%	103%	93%
Class B	-0.35%	0.16%	0.02%	102%	93%
Class C	-0.17%	0.00%	0.08%	103%	96%
Class D	-0.75%	-0.59%	-0.48%	102%	86%
Class E					
<b>Overall (Ref)</b>	-0.35%	-0.13%	-0.14%	103%	92%
Class F (optional)	-0.59%	0.26%	-0.03%	107%	93%

	Low Delay B Main10				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	-0.31%	-0.05%	0.21%	102%	93%
Class C	-0.15%	-0.03%	-0.26%	102%	93%
Class D	-0.51%	-0.62%	-0.08%	101%	93%
Class E	0.16%	0.77%	0.11%	110%	93%
<b>Overall (Ref)</b>	-0.23%	-0.03%	0.00%	103%	93%
Class F (optional)	-0.56%	0.25%	-0.82%	107%	95%

# Conclusions

	Y	U	V	EncT	DecT
	<b>All Intra Main10</b>				
<b>Adapt. Clip.</b>	-0.15%	-0.05%	-0.04%	103%	95%
<b>Adapt. Clip. + Res. Smooth.</b>	-0.17%	-0.06%	-0.07%	104%	95%
	<b>Random Access Main10</b>				
<b>Adapt. Clip.</b>					
<b>Adapt. Clip. + Res. Smooth.</b>	-0.35%	-0.13%	-0.14%	103%	92%
	<b>Low Delay B Main10</b>				
<b>Adapt. Clip.</b>	-0.21%	-0.04%	-0.14%	102%	93%
<b>Adapt. Clip. + Res. Smooth.</b>	-0.23%	-0.03%	0.00%	103%	93%

- Coding performance of JEM can be improved with Adaptive Clipping
- Added complexity (Decoder/Encoder) is negligible
- Proposed modifications allows cleaning/unify the Clipping process code
- Recommendation is to adopt the above technology in JEM.