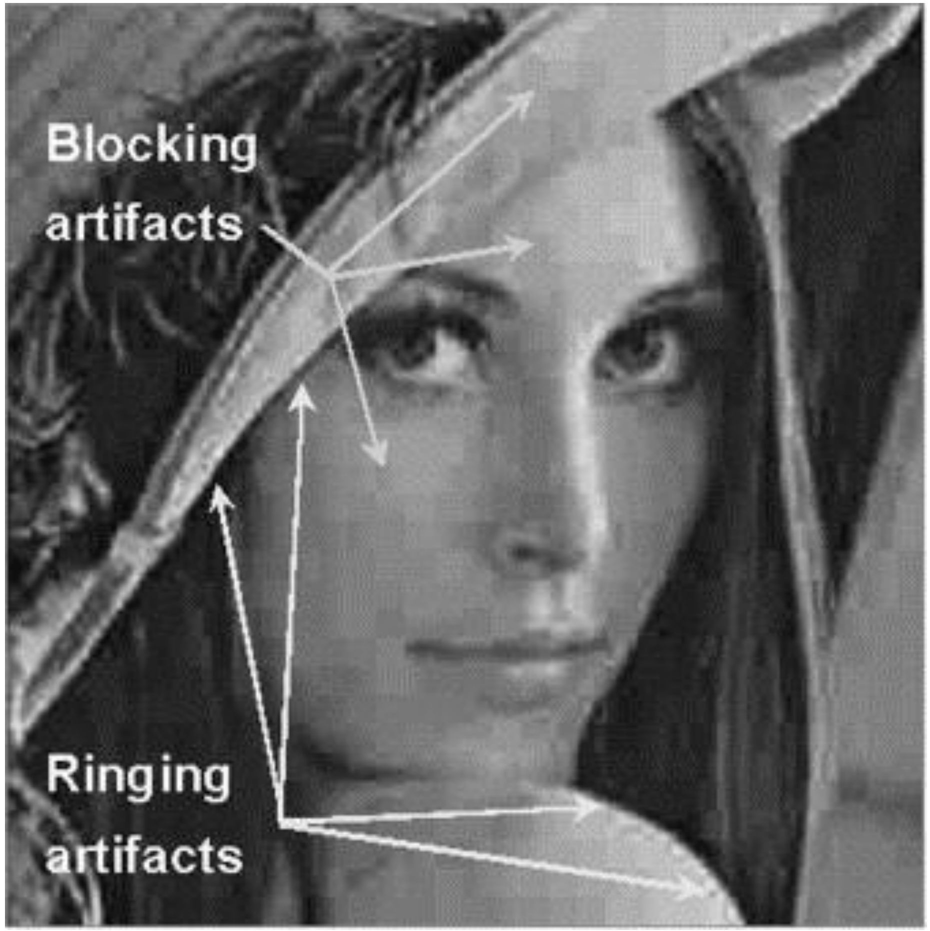
Video Compression

INSTRUCTOR: YAN-TSUNG PENG

DEPT. OF COMPUTER SCIENCE, NCCU

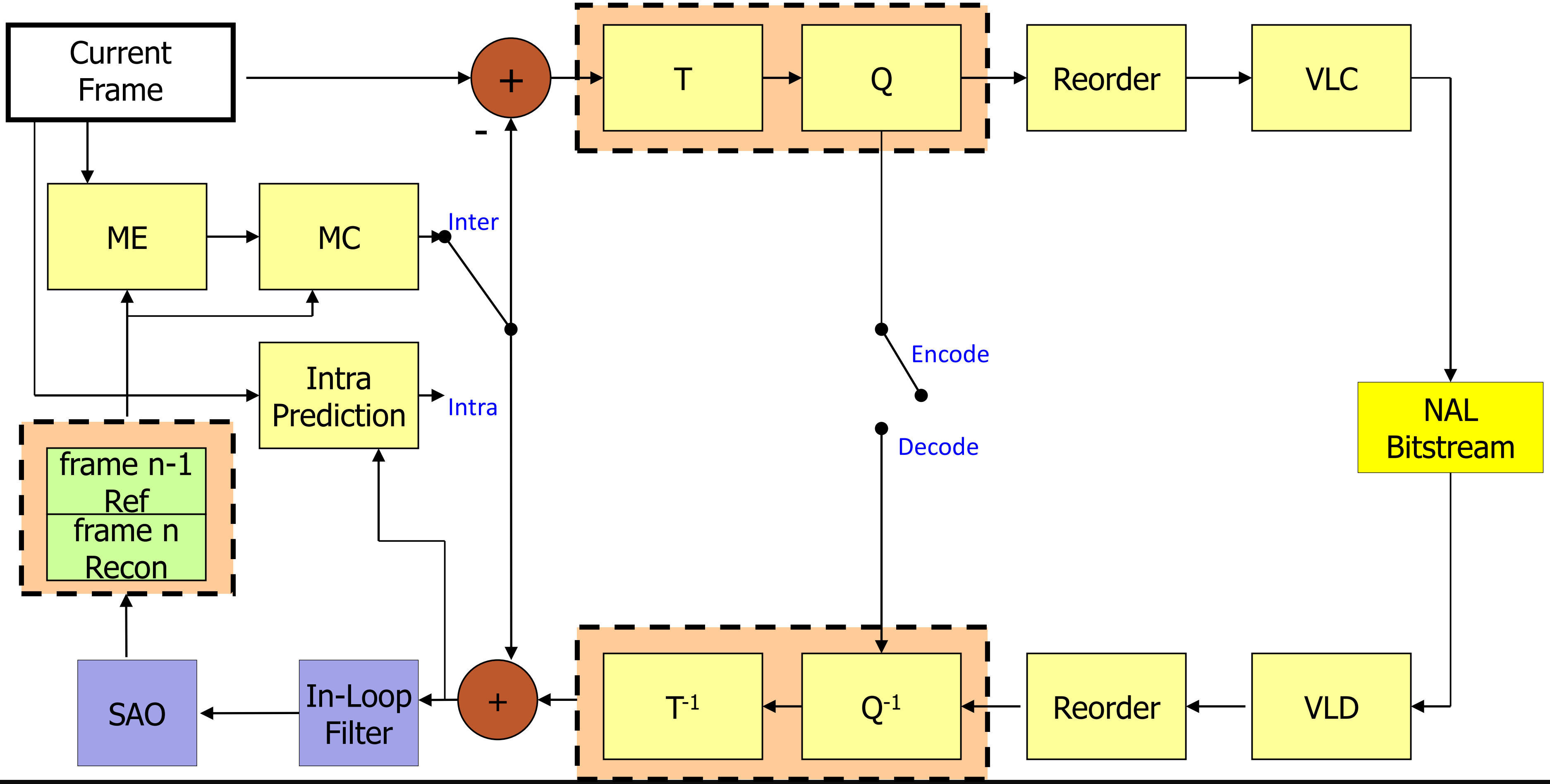
Ringing Effects



- High quantization forces high frequency coefficients to be remove whereas strong edges can be damaged, causing ringing or ripples near the edges

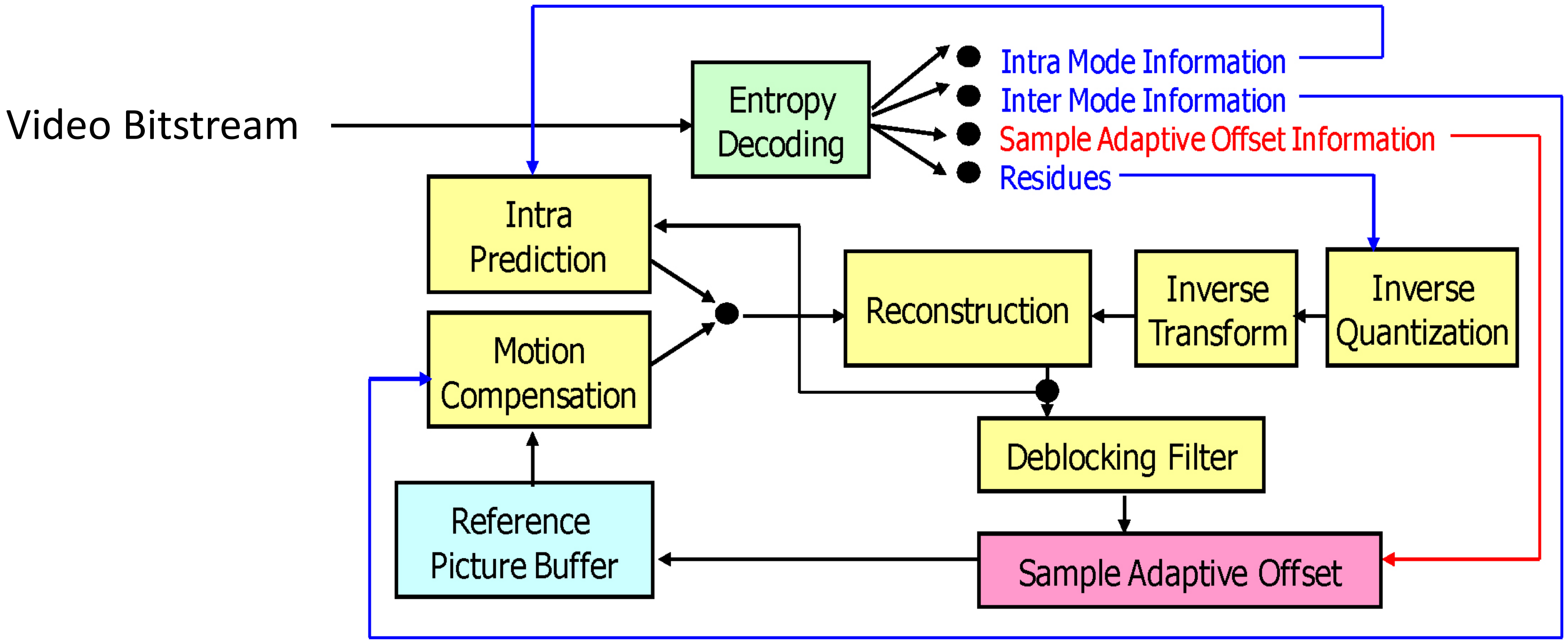
Figure taken from Russo, Fabrizio. "A nonlinear technique based on fuzzy models for the correction of quantization artifacts in image compression." Measurement 32.4 (2002): 273-279.

Sample Adaptive Offset (SAO)



Sample Adaptive Offset

- Another **in-loop filtering technique** added in HEVC



Block diagram of HEVC decoder.

Fu et al. "Sample Adaptive Offset in the HEVC Standard," TCSVT 2012

Motivation and Evolutions

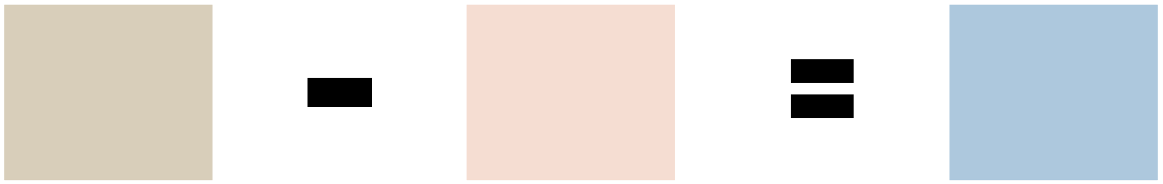
- SAO aims to reduce sample distortion by compensating reconstructed block with differences between decoded and input frames.
- First, it classifies reconstructed samples into various categories. Then, calculating an offset for each category, and adding the offset to each sample of the category.
- Deringing filtering
  - In Windows Media Video 9 (WMV9) (Srinivasan et al., “Windows Media Video 9: Overview and applications,” *Signal Process. Image Commun.*, 2004.)
    - Samples are in two categories: edge or nonedge samples, filtered differently.
  - In VCEG-AL27 (Chien et al., Adaptive Filter Based on Combination of Sum-Modified Laplacian Filter Indexing and Quadtree Partitioning, document VCEG-AL27, Jul., 2009.
    - Reconstructed blocks are classified into different categories using values of sum-modified Laplacian.
    - It shows in-loop filtering can provide better coding efficiency than post-filtering.

SAO Overview

- A technique used in the High-Efficiency Video Coding (HEVC) standard to improve the quality of the reconstructed video by reducing the distortion introduced during the compression process.
- HEVC supports larger transform sizes (4x4 to 32x32) than AVC (up to 8x8). While larger transforms improve energy compaction, they can also cause more artifacts, especially ringing, due to quantization errors.
- SAO works by applying different offsets to the pixel values based on their classification, thereby mitigating the effects of quantization errors.
- SAO is applied after the deblocking filter in the loop filter stage of HEVC.
- The main goal is to reduce artifacts and improve the visual quality of the reconstructed image by classifying the reconstructed pixels into different categories and applying appropriate offsets to each category.

SAO History & Characteristics

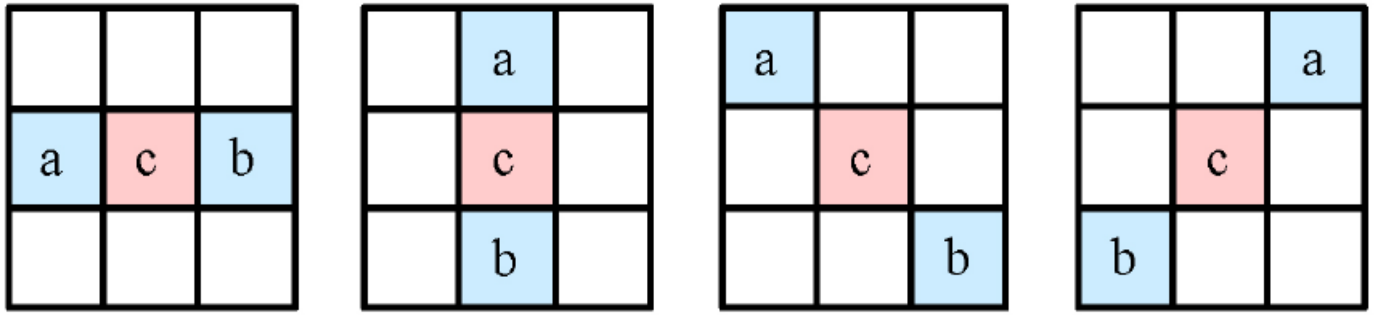
- In JCTVC-E049 (Fu et al.,, CE13: Sample Adaptive Offset with LCU-Independent Decoding, document, 2011.), sample adaptive offset (SAO) was proposed.
- SAO achieves good coding efficiency with reasonable complexity and was added to the working draft of HEVC.
- SAO is applied to luma and chroma in one CTU
  - Note that CTU has a luma coding tree block (CTB) and two chroma CTBs.
- SAO attempts to reduce the **Mean Distortion** between original and reconstructed data.



Original Frame **Decoded Frame** **Difference**

SAO

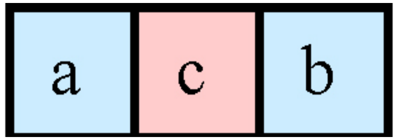
- SAO classifies the pixels into different categories based on their local characteristics.
- SAO has two types in HEVC: edge offset (EO) and band offset (BO).
  - EO: samples classified on comparison between current and neighboring samples (edge direction).
  - BO: samples classified on sample values (in bands, grouping pixels into different bands of intensity)
- Edge Offset
  - EO has four 1-D directional patterns: horizontal, vertical, 135° diagonal, and 45° diagonal.



Horizontal (EO class = 0) | Vertical (EO class = 1) | 135° diagonal (EO class = 2) | 45° diagonal (EO class = 3)

EO in SAO

- For a given EO class, each sample inside the CTB is classified into five categories
  - The current sample “c,” is compared with its two neighbors (“a”, “b”) along the selected 1-D pattern.



  - Five Categories:

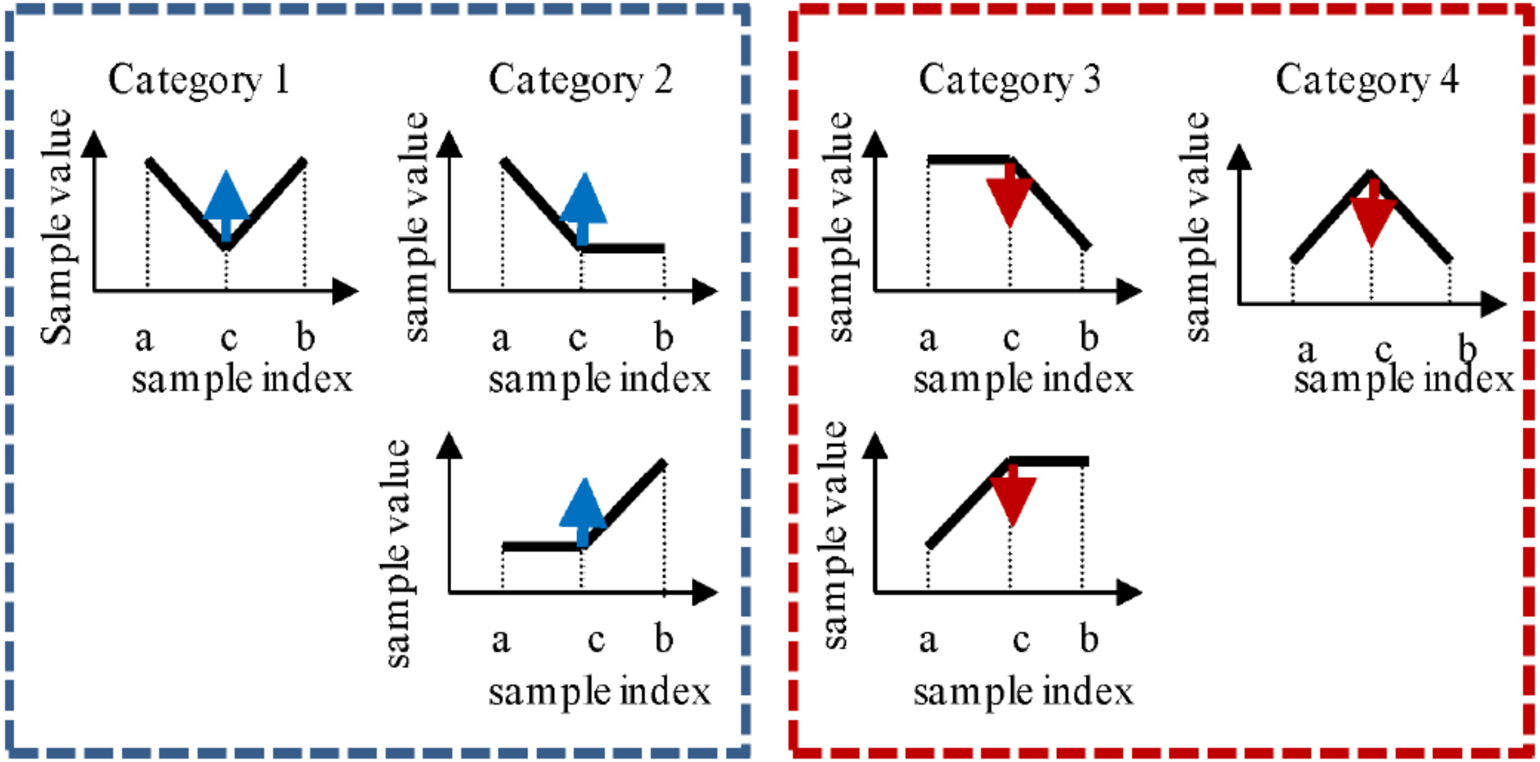
SAMPLE CLASSIFICATION RULES FOR EDGE OFFSET

Category	Condition
1	$c < a \;\&\&\; c < b$
2	$(c < a \;\&\&\; c{==}b) \vert\vert (c == a \;\&\&\; c < b)$
3	$(c > a \;\&\&\; c{==}b) \vert\vert (c == a \;\&\&\; c > b)$
4	$c > a \;\&\&\; c > b$
0	None of the above

EO in SAO

For each category, a comparison is made between the pixel value and its neighboring pixels in the specified direction. Based on the comparison, the pixel is assigned an offset from a predefined set of offsets.

- Positive offsets for categories 1 and 2
  - Smoothing
- negative offsets
  - Sharpening (not allowed, statistical reasons)
- Negative offsets for categories 3 and 4
  - Smoothing
- Positive offsets
  - Sharpening (not allowed)
- **Category 0**
  - No SAO

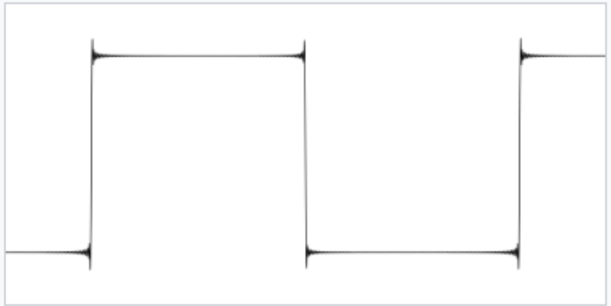


Gibbs Phenomenon

- **Jump discontinuity** will be caused by approximating a periodic function by additions of piecewise continuously differentiable periodic functions, resulting overshoot and undershoot near the discontinuities.

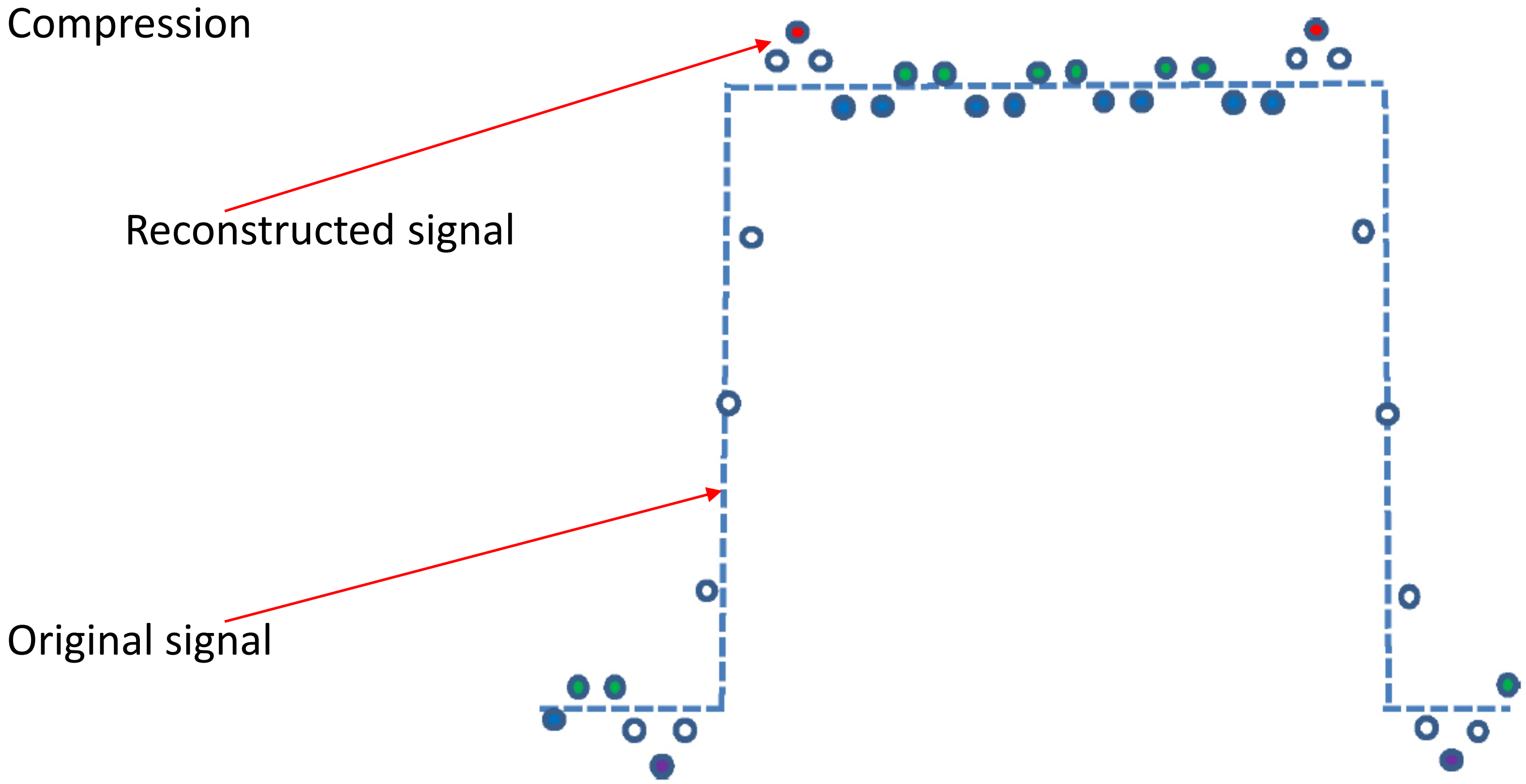
Functional approximation of square wave using 5 harmonics

Functional approximation of square wave using 25 harmonics



Functional approximation of square wave using 125 harmonics

Gibbs Phenomenon



Compression Artifacts

The Gibbs phenomenon causes peaks and valleys near sharp edges when high frequencies are removed, similar to ringing artifacts seen in video compression.

Edge Offset

Applying negative offsets to peaks/convex corners and positive offsets to valleys/concave corners (as in SAO's Edge Offset) helps smooth sharp transitions and reduce Gibbs-related artifacts.

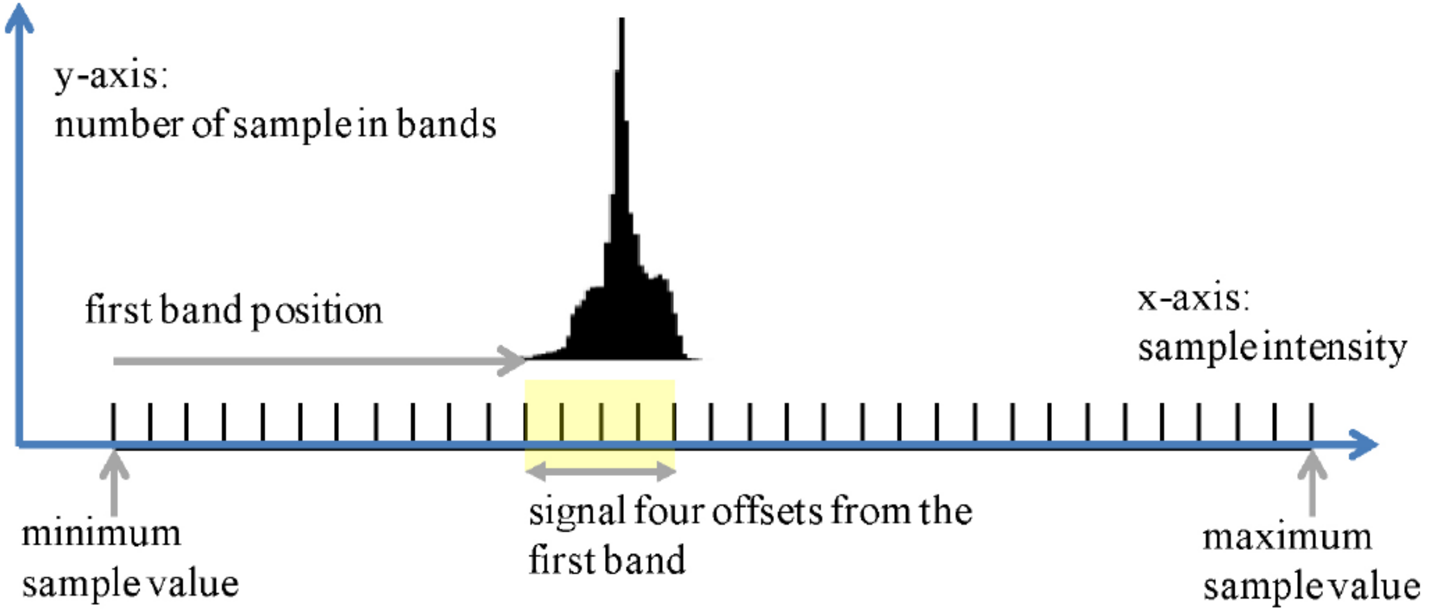
Fu et al. "Sample Adaptive Offset in the HEVC Standard," TCSVT 2012

Band Offset (BO) in SAO

- ❑ The sample value is equally divided into 32 bands.
  - ❑ Ex: 8-bit ranging from 0 to 255.
  - ❑ Bandwidth is 8.
  - ❑ One offset is added to all samples of the same band.
- ❑ For a CTB, using offsets of four consecutive bands with the starting band position transmitted to the decoder

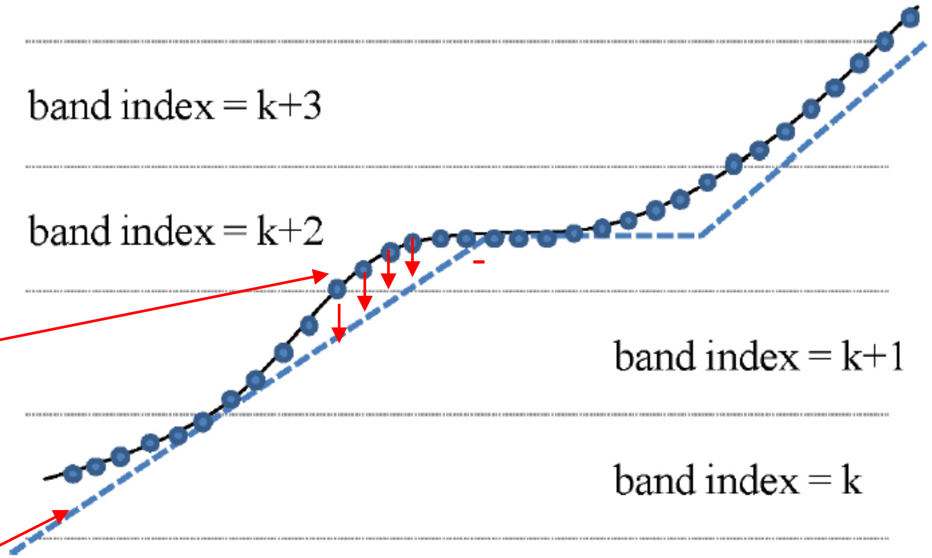
Pixels are classified based on which band their intensity value falls into, and a corresponding offset is applied to the pixels in each band.

When pixel intensities fall into these four bands, offsets will be added to these intensities.



Fu et al. "Sample Adaptive Offset in the HEVC Standard," TCSVT 2012

BO



Reconstructed signal

Original signal

The reconstructed signals are shifted to the left of the original signals, which can be corrected by negative values of BO for bands k, k + 1, k + 2, and k + 3.

The average difference between the original and reconstructed signals in a band is signaled to the decoder. (no constraint on offset signs).

SAO Parameters and Encoding

- Offsets
  - For each category (EO or BO), SAO defines a set of offsets that are used to adjust the pixel values. These offsets are signaled in the bitstream for each coding tree unit (CTU).
  - The encoder determines the optimal offsets by minimizing the distortion between the original and reconstructed pixel values.
- Flagging
  - For each CTU, the encoder signals whether SAO is applied and the type of SAO (EO or BO). If SAO is applied, the specific offsets for each category or band are also signaled.
- SAO Application in Decoding
  - During decoding, the decoder reads the SAO parameters from the bitstream, including the type of SAO (EO or BO), the offsets, and the classification information.
  - The decoder applies **the offsets** to the reconstructed pixel values based on the classification information. This process adjusts the pixel values to reduce the quantization distortion and improve the overall video quality.

SAO Workflow

1. **Original and Reconstructed Image**: The encoder first processes the original image and produces a reconstructed image after applying deblocking.

2. **Classification**: The encoder classifies the reconstructed pixels into different categories (EO or BO).

3. **Offset Calculation**: The encoder calculates the optimal offsets for each category.

4. **Bitstream Signaling**: The encoder signals the SAO parameters in the bitstream.

5. **Decoding**: The decoder reads the SAO parameters and applies the corresponding offsets to the reconstructed image.

6. **Final Result**: The decoder produces the final frame with reduced artifacts and improved quality.

Summary

❑ SAO Mode Signaling

For each CTU, the encoder signals the SAO type: OFF, Edge Offset (EO), or Band Offset (BO). EO includes a class (0–3); BO includes a start band (0–28) and four offsets. Chroma (Cb, Cr) shares the SAO type and EO class to reduce bit cost.

EO Example

- EO

Category = 1

Diff=2

10	7	4	5
11	8	6	4
11	10	12	10
11	12	13	12

Decoded blk

10	8	3	5
12	10	7	6
12	13	11	11
12	12	12	12

Orig blk

Category = 1

Dec	Ori	Offset
8	10	2
6	7	1
10	13	3

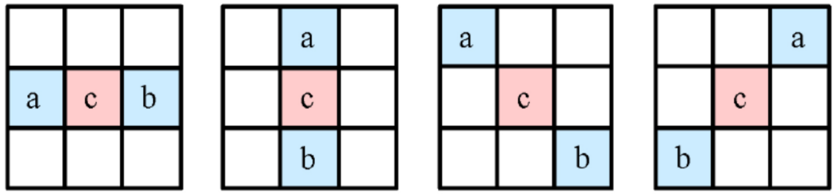
Assigned Offset = (2+1+3)/3=2

Category = 3

Dec	Ori	Offset
12	11	-1

Assigned Offset = -1

EO class = 2 (encoded)



Horizontal (EO class = 0) | Vertical (EO class = 1) | 135° diagonal (EO class = 2) | 45° diagonal (EO class = 3)

BO Example

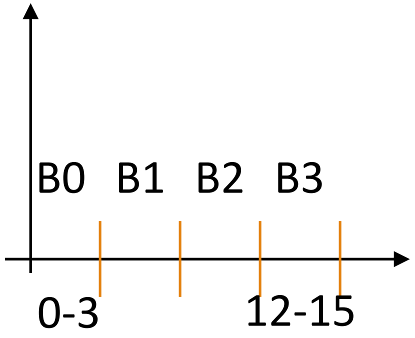
- BO

10	7	2	5
11	8	6	1
11	10	12	10
11	12	13	12

Decoded blk

10	8	3	5
12	10	3	6
12	8	11	11
12	12	12	12

Orig blk



	0-3	4-7	8-11	12-15
	B0	B1	B2	B3
Dec	(2+1)/2=1.5	(5+6+7)/3=6		
Ori	(3+6)/2	(5+3+8)/3		
Oft	3	-0.7		

Benefits of SAO

- Quality Improvement
  - SAO significantly improves the subjective and objective quality of the reconstructed video by reducing artifacts such as blocking and banding.

- Adaptivity
  - The adaptive nature of SAO allows it to apply different offsets to different regions of the image, making it effective in a wide range of video content and scenes.

SAO Results

Before

After

SAO Results

Before

After

Rate Distortion Optimization

Rate Distortion Optimization (RDO)

- Balance between the amount of *distortion and* the bits required to encode the video (*rate*).
- The rate-distortion efficiency for video compression can be increased based on a sophisticated combination among various coding decisions.
- As for coding decisions, for example:
  - How to segment a frame into blocks?
  - Which mode (intra or inter) a block should be coded?
  - What MV should an inter block use versus what intra-prediction mode an intra block use?
  - How to encode a block from the point of view of entropy coding?
- To address video encoding optimization, a lot of research done for discussing rate-distortion optimization of video compression

G. Sullivan and T. Wiegand, Rate-distortion optimization for video compression, IEEE Signal Process. Mag., November 1998.

RDO

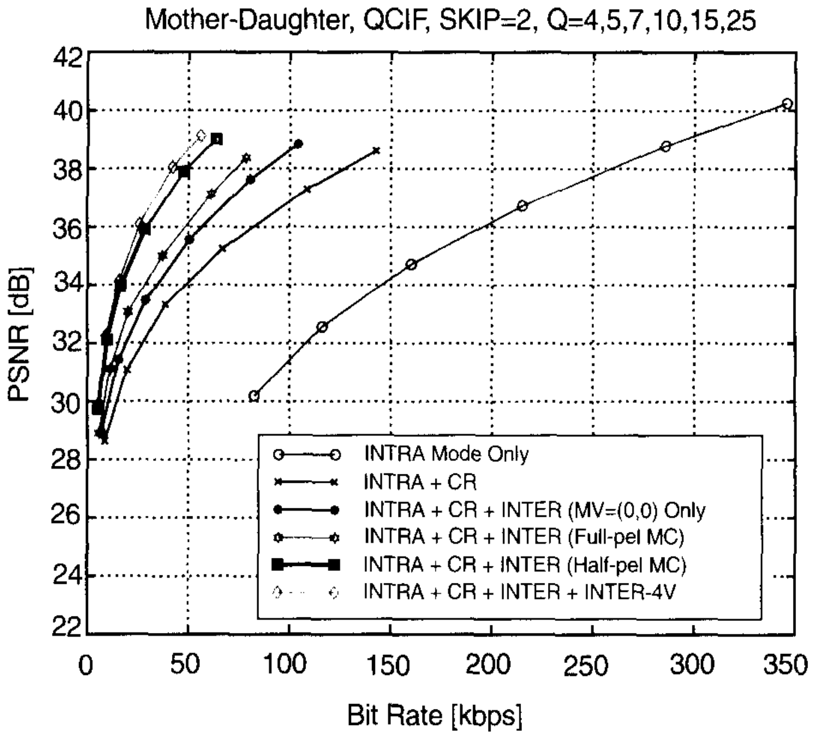
- RDO usually is discussed and represented using Largrangian methods as

Lagrangian rate-distortion function:

$$\text{Minimize } J, \qquad s.t. J = D + \lambda R$$

- D represents a distortion measure, which could be
  - SSD (sum of square difference), MSE, SAD, etc.
- R represents the number of bits used

SKIP mode: A prediction mode which uses a block copied directly from the collocated block in a previously decoded frame.



CR means SKIP or Intra

RDO

- It can have various forms, such as

$$\operatorname{argmin}_{M,Q} J_A(M,Q) = D_A(M,Q) + \lambda R_A(M,Q),$$

where $M \in \{INTRA, SKIP, INTER\}$,

$Q: set\ of\ selected\ quantization\ steps,$

$A: set\ of\ selected\ blocks$

Video Coding Standard – H.264, H.265

Video Coding Standard – H.264

- Developed by Joint Video Team (JVT) of MPEG and ITU-T VCEG
- It has a three-step design
  - Video Coding Layer (VCL):
    - Source video
  - Network Abstraction Layer (NAL):
    - Bitstream data error-resiliently
  - Transport Encapsulation Layer (TEL):
    - NAL units over real transport systems

Video Coding Layer

- Predictive coding
  - 13-mode luma, 4-mode chroma intra prediction
- 16-bit integer combined transform/quantization
  - Exact forward-inverse transform pair is used
  - Transform block size is 4x4
- Two entropy coding methods
  - Universal VLC and Context Adaptive VLC
  - Context Adaptive Binary Arithmetic Coding
- In-loop filter

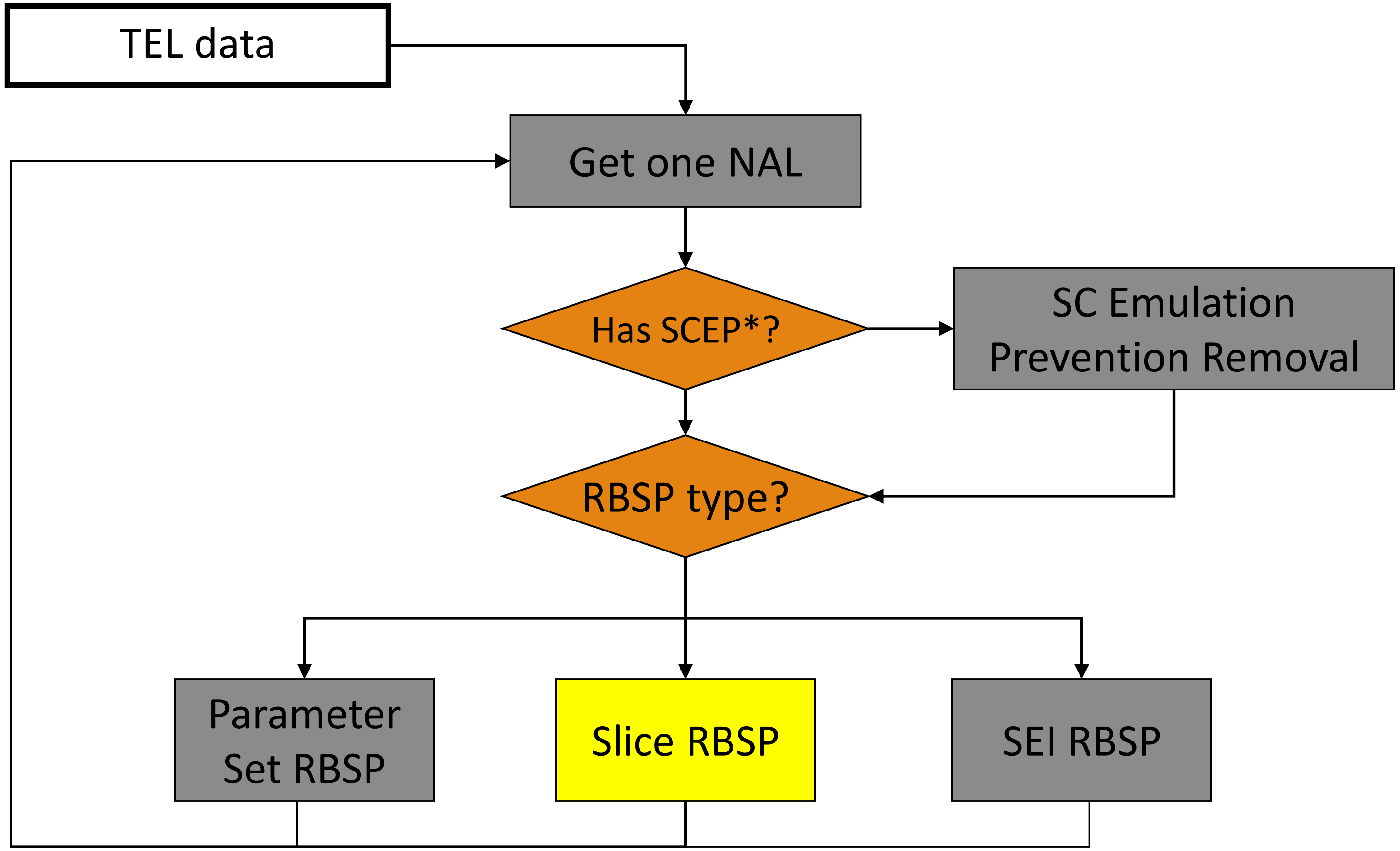
Network Abstraction Layer

- NAL is mainly designed for error-resilience with an packet-based format
- NAL works for both packet-oriented and bitstream-oriented transports
- Each NAL has video and system data
- In bitstream-oriented transports, each NAL can be preceded by a start code prefix
  - Which requires a *start code emulation prevention* mechanism

NAL Units

- A NAL unit contains a header and a Raw Byte Sequence Payload (RBSP)
- NAL unit (RBSP) types
  - Type 1~5: different slices
  - Type 6: Supplemental Enhancement Information (SEI)
  - Type 7: Sequence Parameter Set (SPS)
  - Type 8: Picture Parameter Set (PPS)
  - Type 9: Picture Delimiter (PD)
  - Type 10: Filler Data (FD)
  - 0, 11~31 are reserved or for external use

NAL Decoder Flowchart



*SCEP: Start Code Emulation Prevention

SCEP

- Emulation prevention byte pattern
  - 0000 0011
- If not in a header, the pattern, “0000 0000 0000 0000”, emulation prevention byte pattern needs to be inserted.

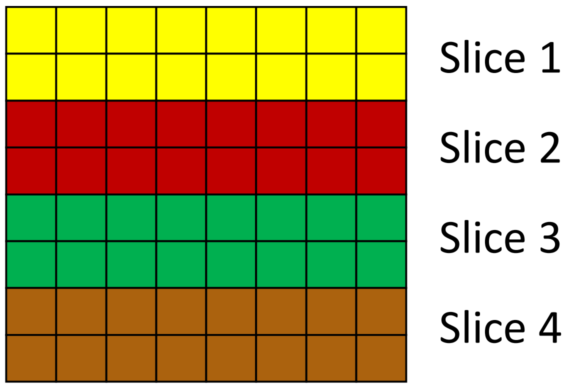
  0000 0000 0000 0000 0000 00xx (binary)

  0000 0000 0000 0000 0000 0011 0000 00xx

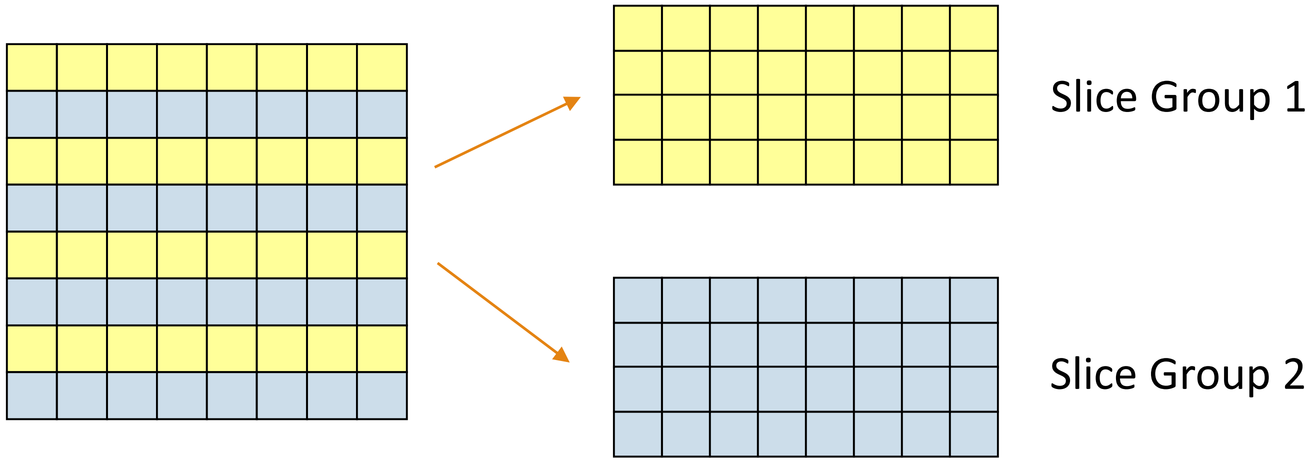
- The decoder inverse the above process to restore the original RBSP

Slice Mode

- Macroblocks can be assigned to different slices or to different slice groups



Normal Slice Mode

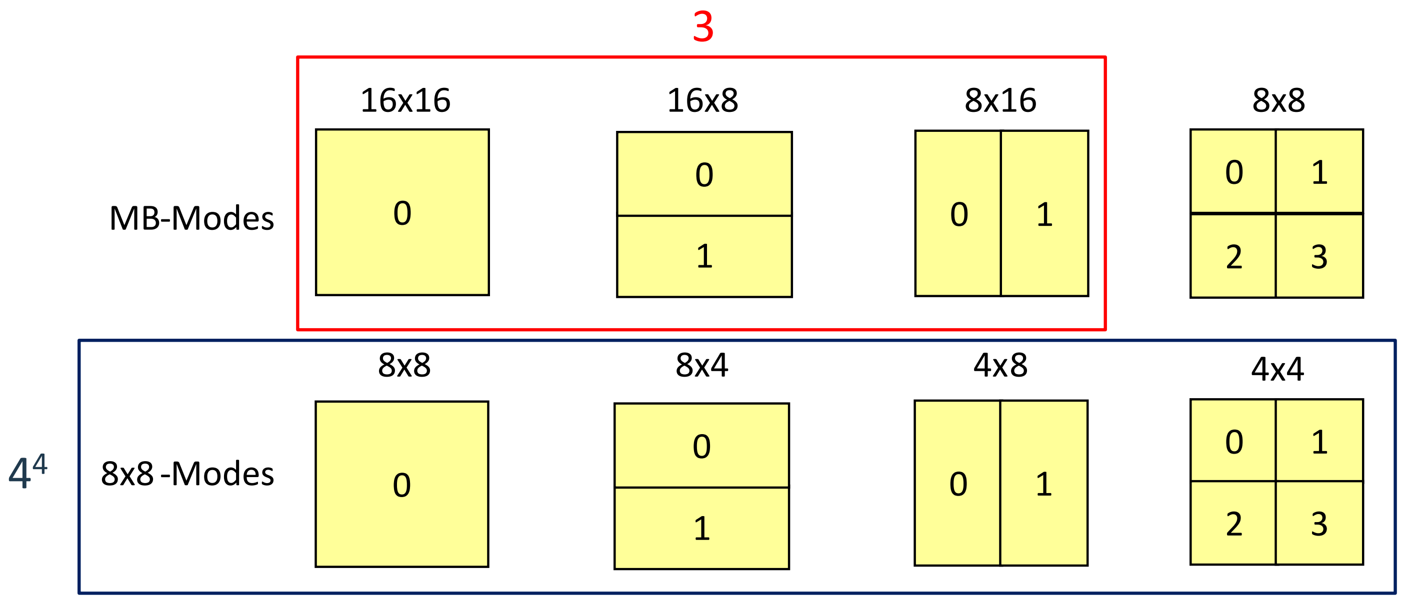


Flexible Macroblock Ordering

Each slice group is further divided into slices

Coding Units

- Quadtree partition
  - H.264 has $3+4^4$ possible partition modes:



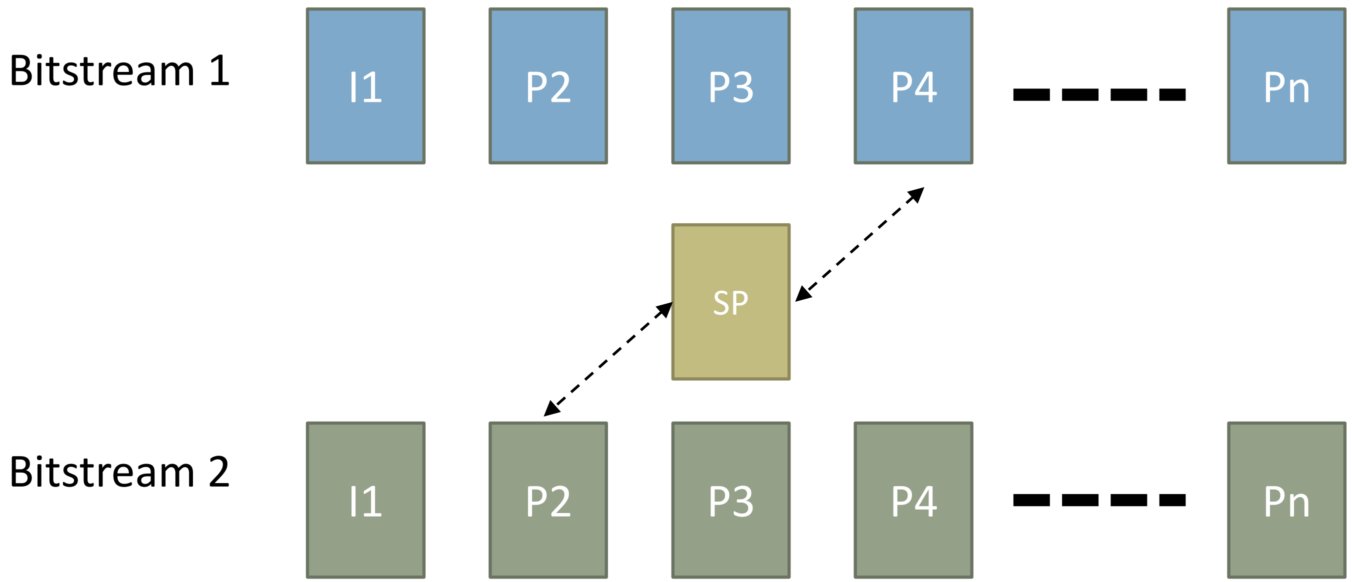
Frame, Slice, MB Types

- Frame types
  - I, P, B frames
  - I-frames contain only intra macroblocks
  - P-frames contain either intra macroblocks or predicted macroblocks
  - B-frames contain intra, predicted, or bi-predicted macrobocks
- 5 slice types
  - I-slice, P-slice, B-slice, SI-slice, SP-slice
  - SI-slices (Switching I) contain SI-macroblocks (intra coded macroblocks)
  - SP-slices (Switching P) contain P and/or I-macroblocks

Switching between Bitstreams using SI/SP Frames

A server should allow these variations:

Bandwidth variations available to a user



If we switch Bitstream 2 to Bitstream 1, P4 of Bisteam 1 will not be decoded correctly.

SP/SP (or SP/SI) pairs are served as entry points into the new stream

Introduction to H.265

- High Efficiency Video Coding (HEVC) standard has been recently developed jointly by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) standardization organizations
- HEVC generally has the same basic structure as H.264/AVC
- Still, it has many incremental improvements:
  - Provide more flexible block partitioning sizes
  - Introduce the quadtree decomposition and structure for PUs and TUs
  - Provide more prediction modes
  - Support efficient parallel processing
    - A frame can be divided into a grid of rectangular regions that can independently be decoded/encoded.

Performance compared to H.264

Coding Standard	Average bit rate reduction compared to H.264			
	480p	720p	1080p	2160p
H.265	52%	56%	62%	64%

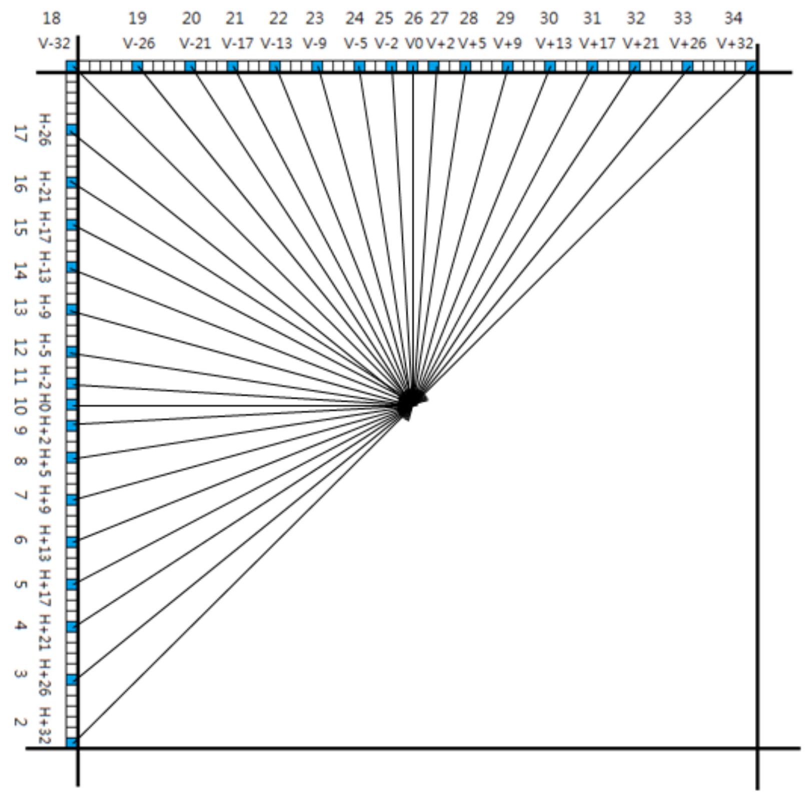
Based on the same subjective video quality

Used reference software: The HM-12.1 HEVC encoder versus the JM-18.5 H.264/MPEG-4 AVC encoder

Ref: TK Tan; Marta Mrak; Vittorio Baroncini; Naeem Ramzan (2014-05-18). "Report on HEVC compression performance verification testing". JCT-VC. Retrieved 2014-05-25.

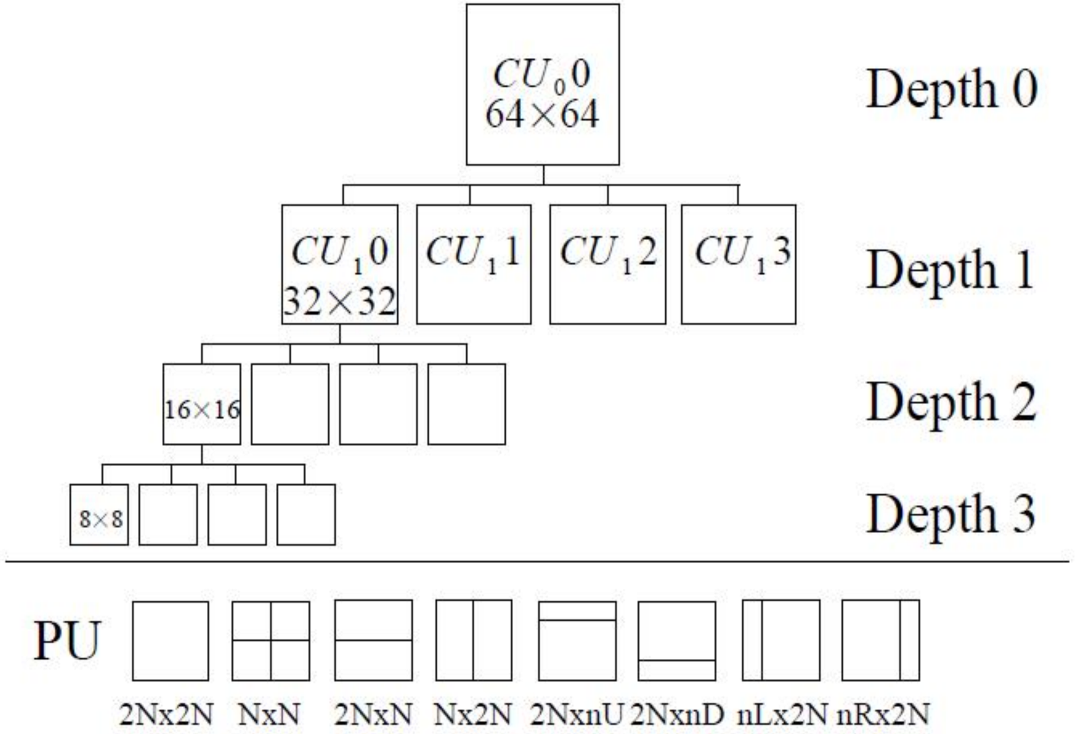
More Intra Prediction Modes

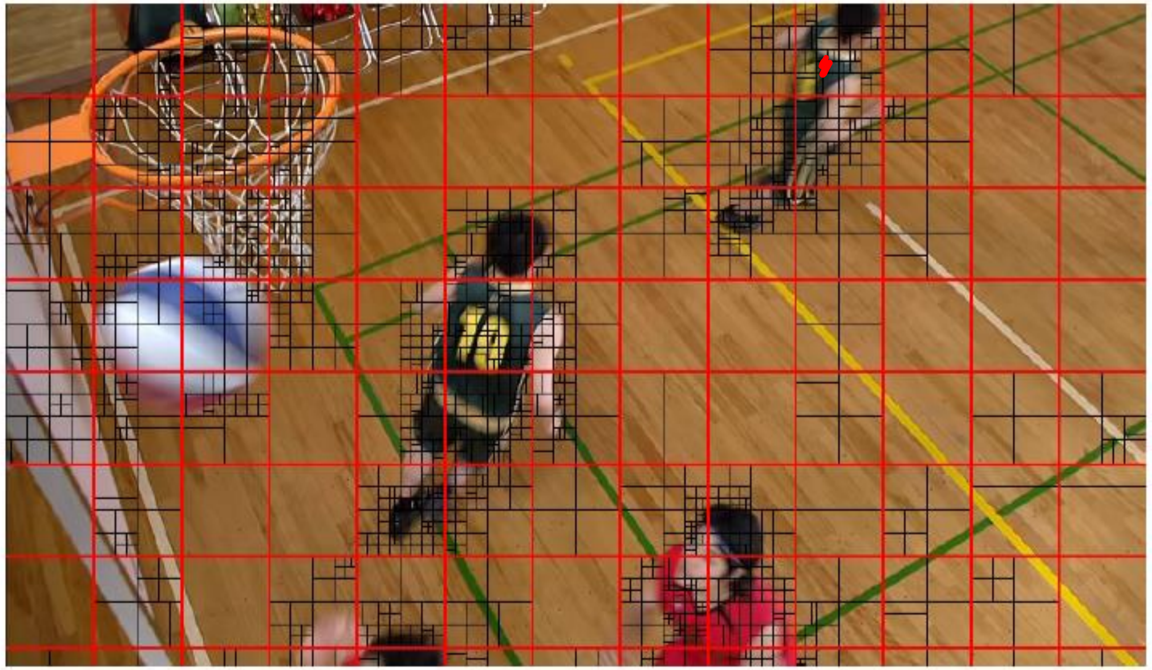
- HEVC has 33 directional modes for intra prediction while H.264 has only 8 directional modes for intra prediction specified by H.264/MPEG-4 AVC.
  - Excluding the DC mode.



Picture is taken from
https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding#/media/File:HEVC_angular_intra_prediction_modes.png

Flexible Block Partitions

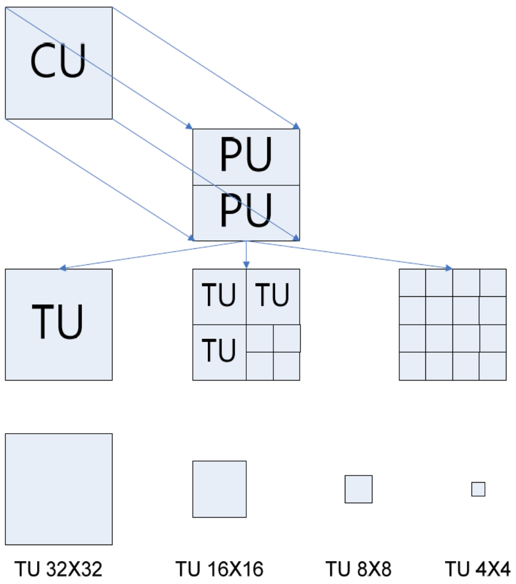


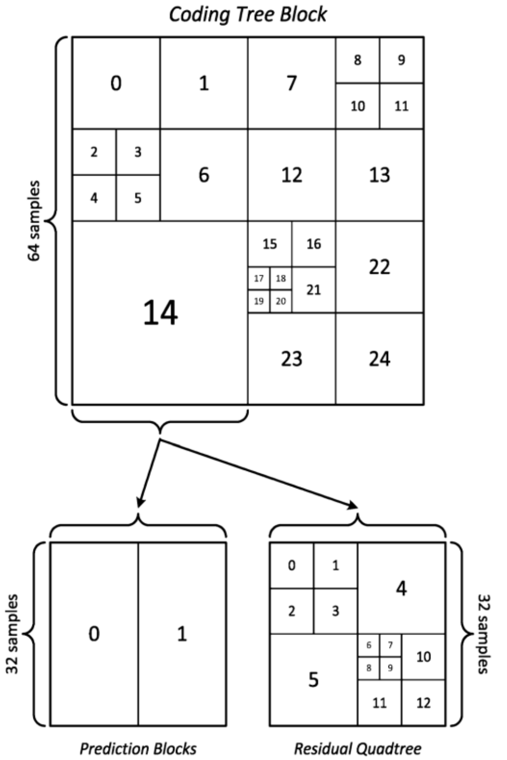


Block Partition Sizes

- Adopted first by H.265/HEVC, a **Quadtree Structure** is designed for subdividing a frame into different block sizes for prediction and residual coding
  - Prediction block sizes range from 4x4 to 64x64
  - Transform block sizes range from 4x4 to 32x32

Video coding standard	Supported block sizes for motion-compensated prediction
H.262 | MPEG-2 Video	16×16
H.263	16×16, 8×8
MPEG-4 Visual	16×16, 8×8
H.264 | MPEG-4 AVC	16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4
HEVC	64×64, 64×48, 64×32, 64×16, 48×64, 32×64, 16×64, 32×32, 32×24, 32×16, 32×8, 24×32, 16×32, 8×32, 16×16, 16×12, 16×8, 16×4, 12×16, 8×16, 4×16, 8×8, 8×4, 4×8



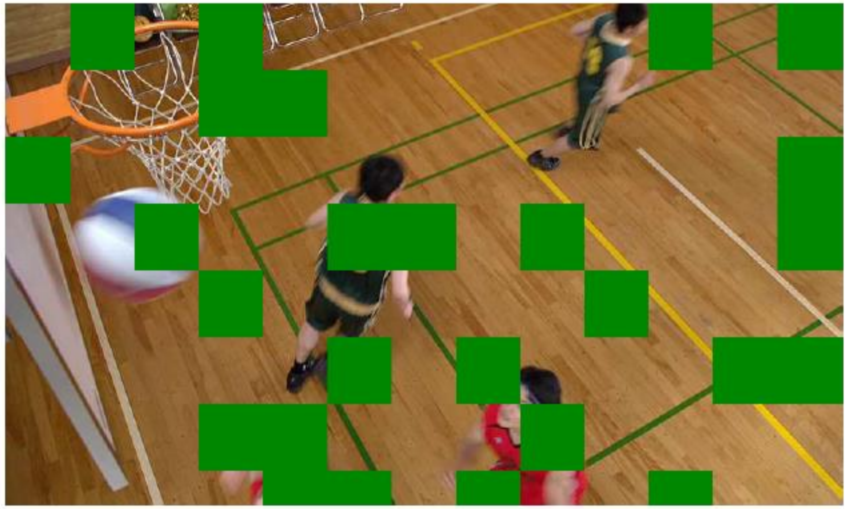


Reference and figures Nguyen et al., Transform Coding Techniques in HEVC, *IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING*, 2013

Introduction to Video Error Concealment

- Packets may be lost happens for various reasons when video is transmitted through an unstable network
  - network congestion, delay, limited bandwidth, etc.

Original Frame



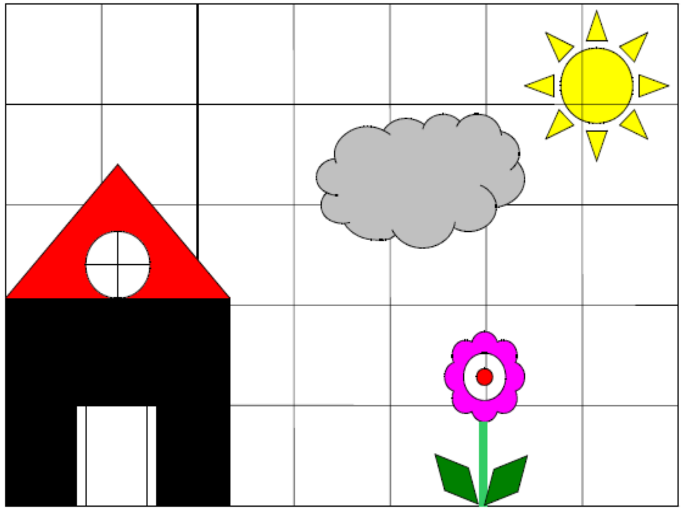
Frame with packet loss

Common Methods for Video Error Concealment

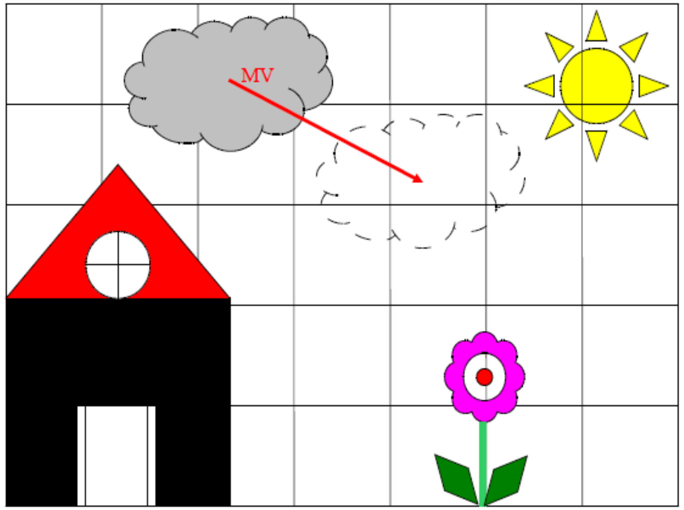
- Spatial Interpolation
  - interpolating lost/corrupted blocks by pixel interpolation from neighboring macroblocks
- Directly copy
  - Conceal an erroneous block by directly copying the collocated one in a reference frame
- Motion compensated error concealment
  - Using those MVs from the spatial and temporal neighbors of a lost block as candidate MVs
  - Finding the most similar block using a boundary matching algorithm (BMA) based on a similarity measurement.
    - Similarity measurement could be MAE, MSE, and SAD, etc.

Boundary Matching Algorithm

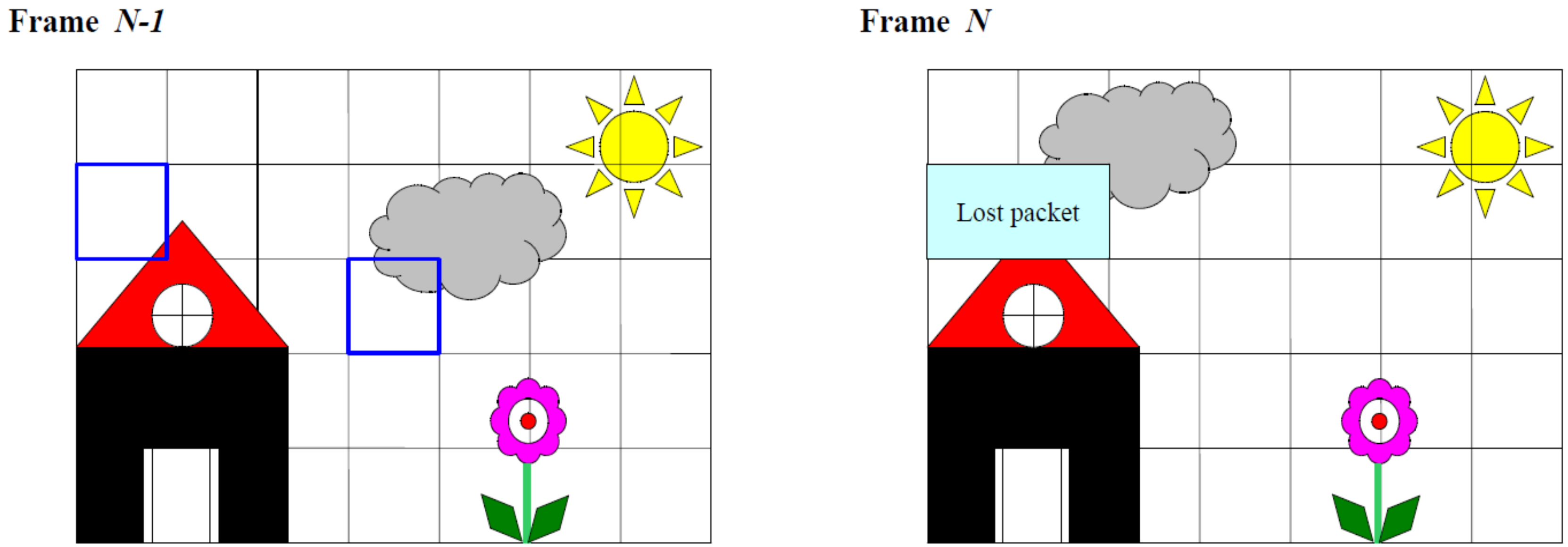
Frame *N-1*



Frame *N*

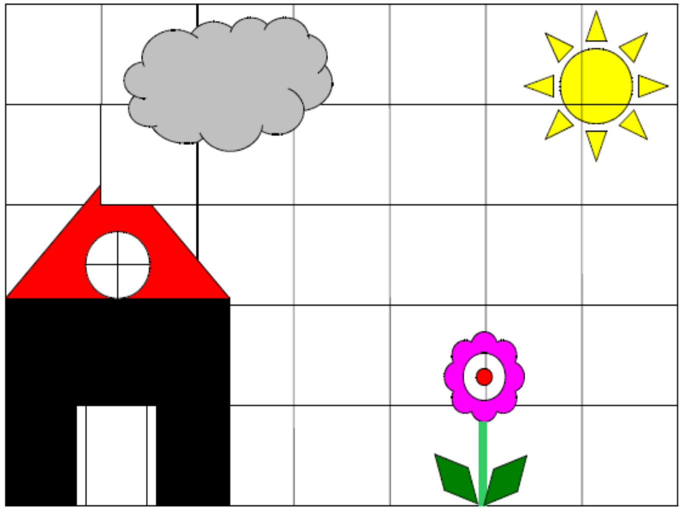


Boundary Matching Algorithm

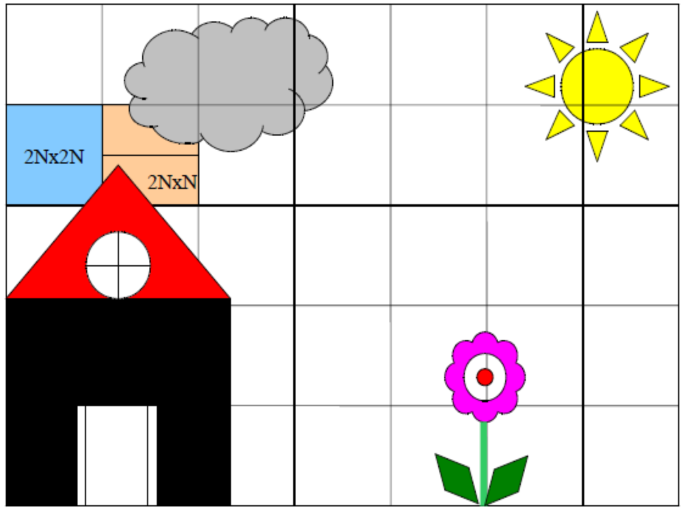


Weighted BMA (WBMA)

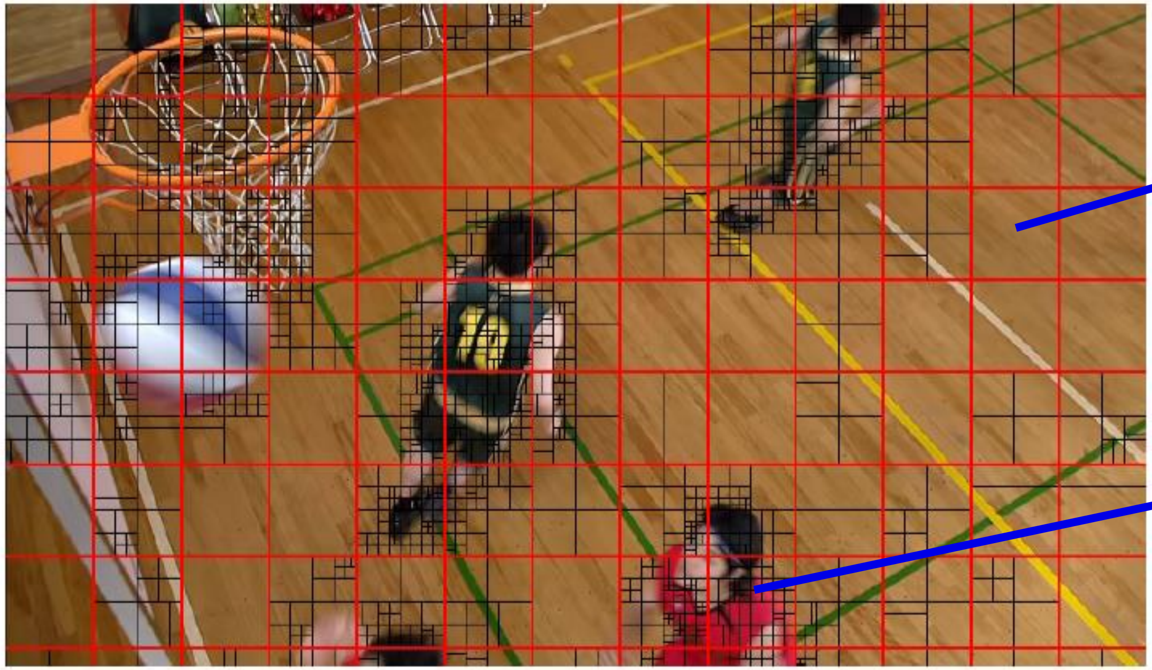
Recovered Frame N



Video Frame N



Characteristics of Block Partitions



Larger CU for a smoother region

Smaller CU for a complex region

Weight Boundary Matching Algorithm (WBMA)

Block Partition Decisions

Use the co-located CU depths and PU partitions of the lost CUs

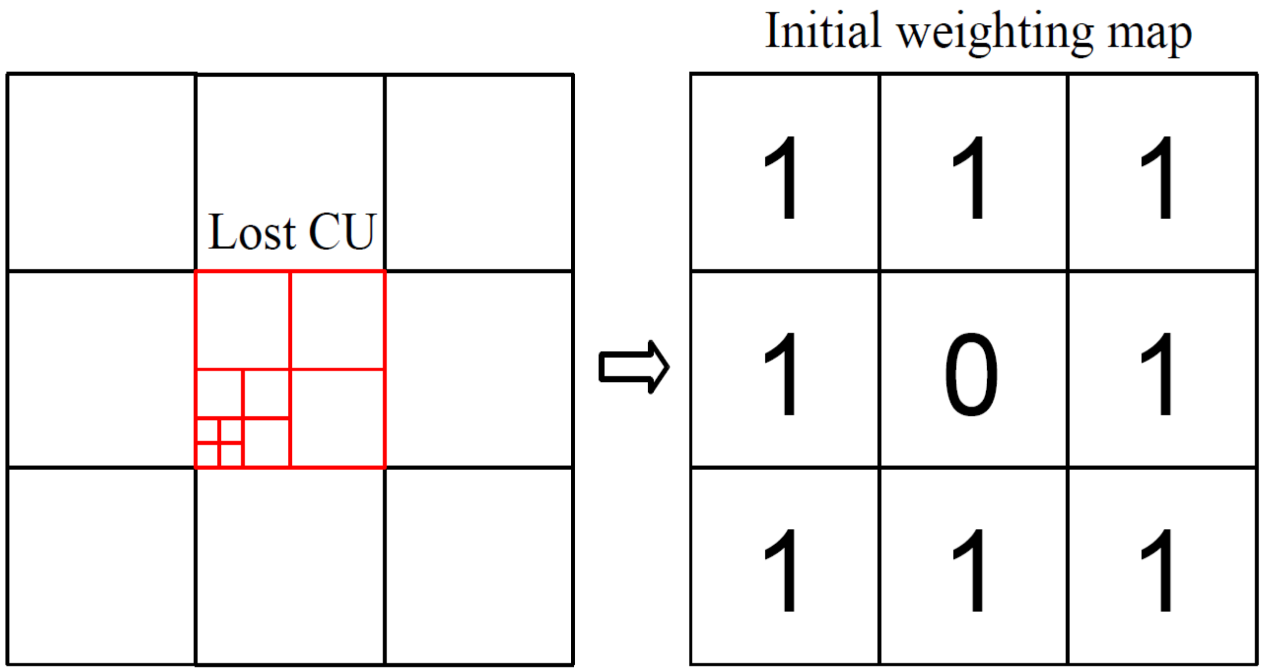
Weighted BMA (WBMA) Steps

1. Construct an initial weighting map
2. Calculate the total weight for lost PUs
3. Select the one with the largest weight to recover

Peng et al., "Weighted Boundary Matching Error Concealment for HEVC Using Block Partition Decisions," Asilomar 2014

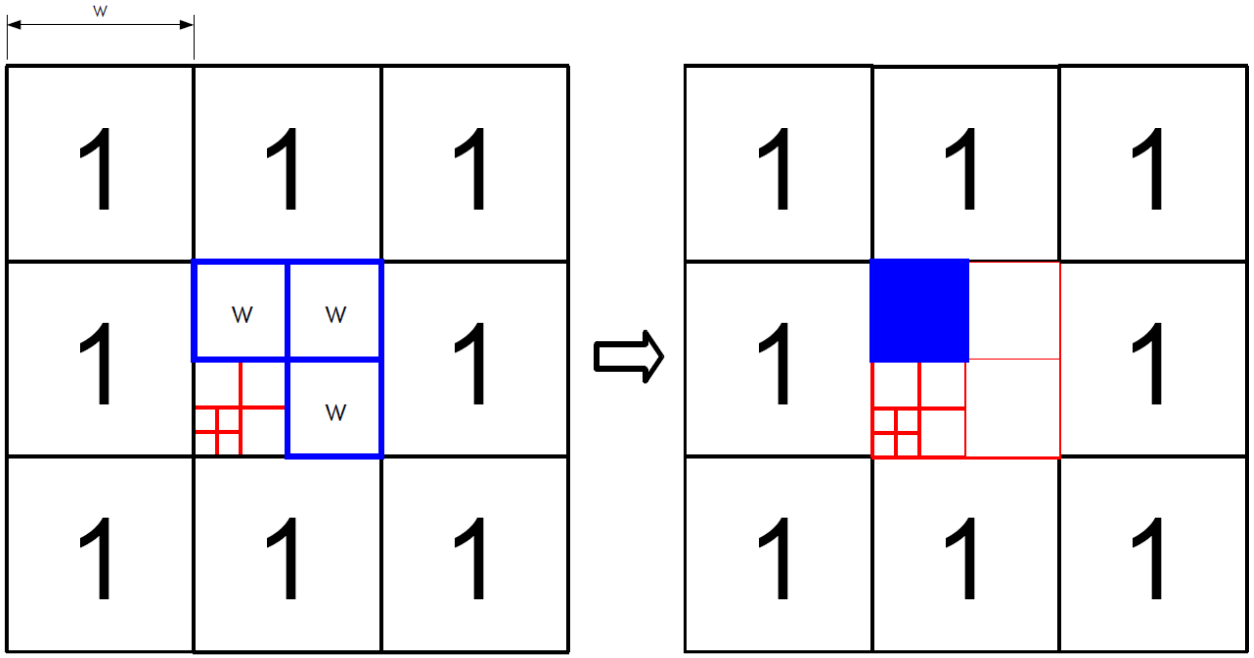
WBMA

Initialize a weighting map



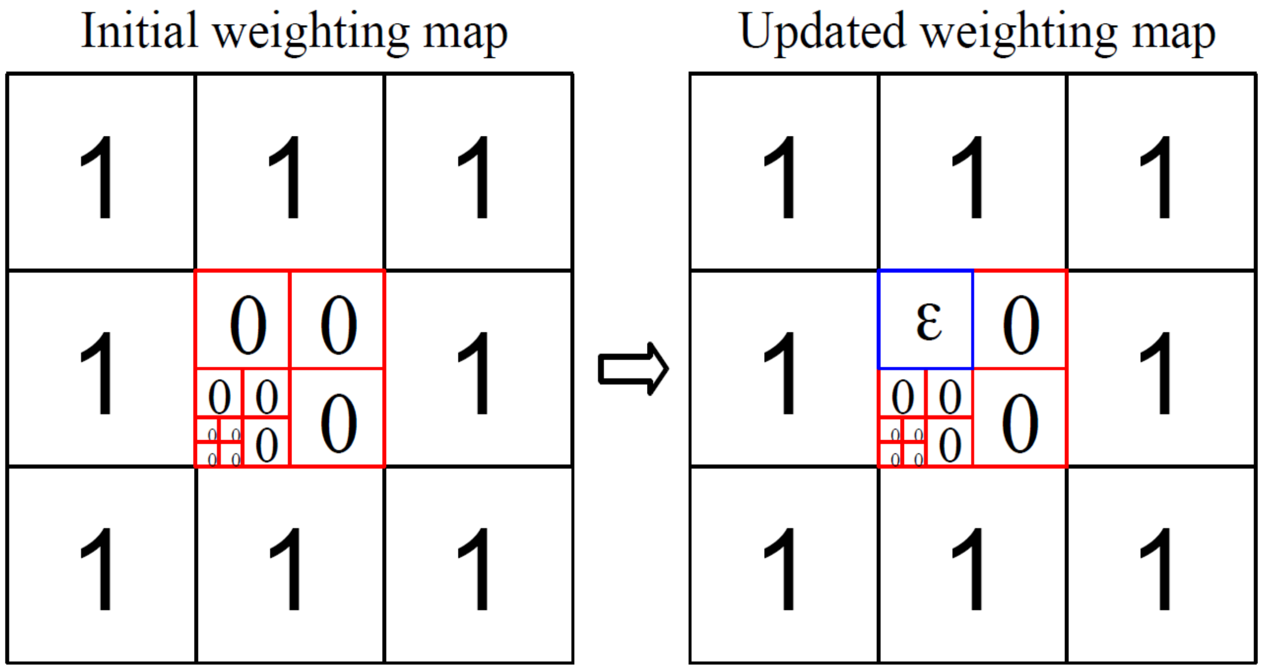
WBMA

Pick the PU with the largest weight to recover



WBMA

Update the weighting map



WBMA Cost Function

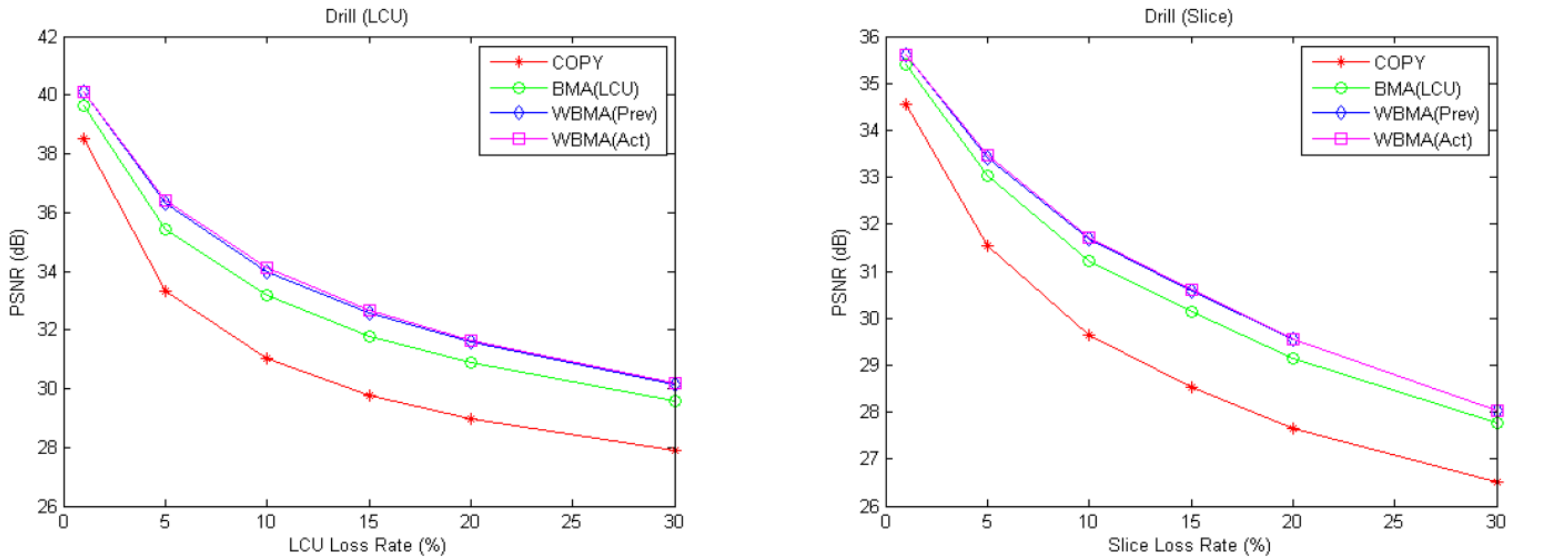
Let F be a frame and W be a weighting map.
For a $l_k \times h_k$ PU_k,

$$
\begin{aligned}
Cost_{PU_k} = \frac{1}{W_{PU_k}} \times [&\sum_{x=x_0}^{x_0+l_k-1} W_{x,y_0-1} \times (|\, F_{x,y_0-1} - F'_{x,y_0-1} \,|) \\
+ &\sum_{x=x_0}^{x_0+l_k-1} W_{x,y_0+h_k} \times (|\, F_{x,y_0+h_k} - F'_{x,y_0+h_k} \,|) \\
+ &\sum_{y=y_0}^{y_0+h_k-1} W_{x_0-1,y} \times (|\, F_{x_0-1,y} - F'_{x_0-1,y} \,|) \\
+ &\sum_{y=y_0}^{y_0+h_k-1} W_{x_0+l_k,y} \times (|\, F_{x_0+l_k,y} - F'_{x_0+l_k,y} \,|)],
\end{aligned}
$$

where $F'_{x,y} = F_{x+MVx,y+MVy}$, W_{PU_k} is the total weight, and MV is the candidate MV

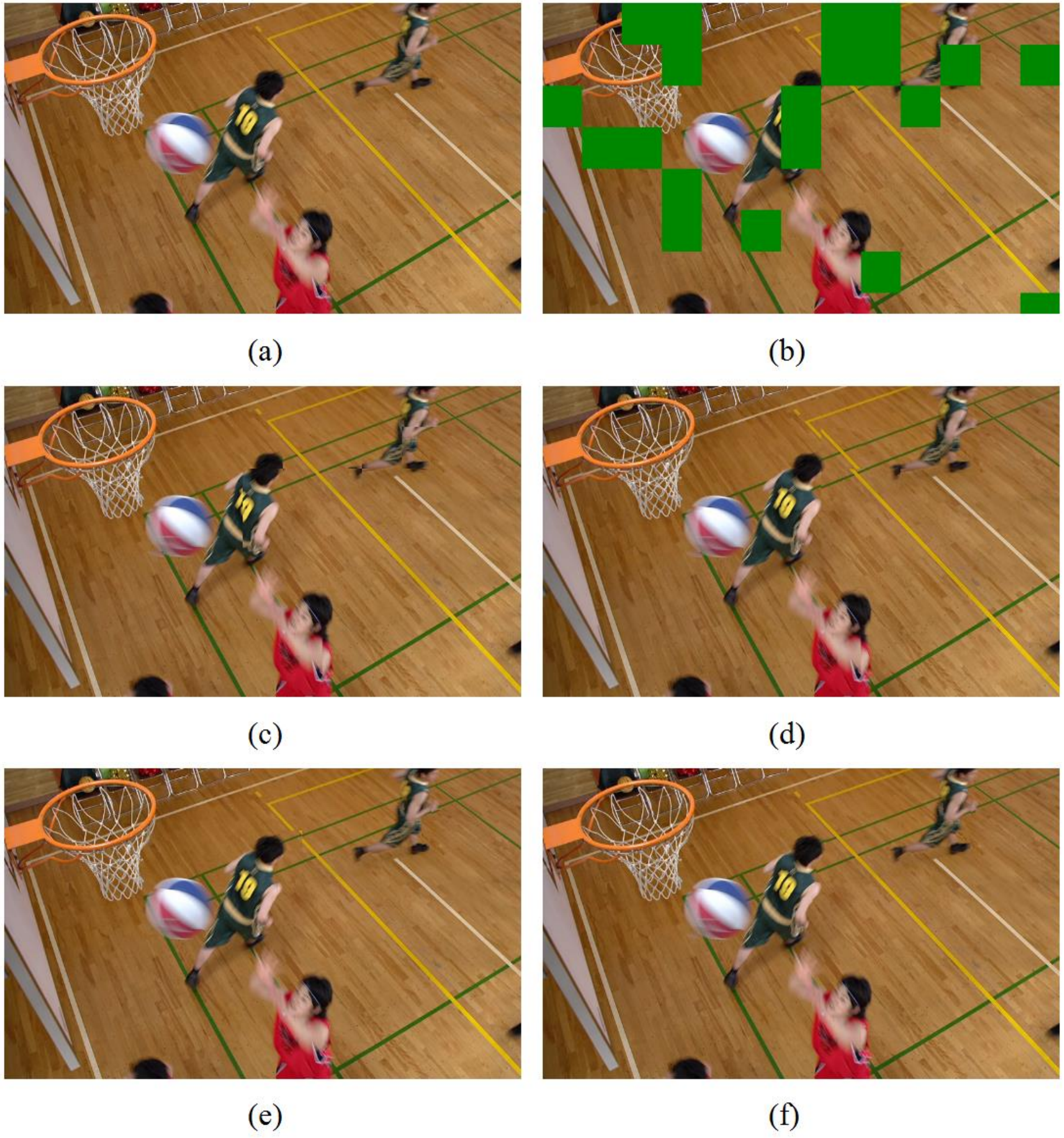
Experimental Results

100 error bitstreams randomly generated for each (LCU means 64×64 CU)



Peng et al., "Weighted Boundary Matching Error Concealment for HEVC Using Block Partition Decisions," Asilomar 2014

Experimental Results



Peng et al., “Weighted Boundary Matching Error Concealment for HEVC Using Block Partition Decisions,” Asilomar 2014