# More Probability Estimators for CABAC in Versatile Video Coding

Varenya Upadhyaya

IITH

June 7, 2021

# About the paper

## Authors

- Sio-Kei Im
- Ka-Hou Chan

## Institute

- Macao Polytechnic Institute

## Time of Publishing

- October, 2020

# Video Coding Format

The compression of a digital video is usually done using a video coding format:

- A V.C.F. is a content representation format for storage or transmission of digital video content.
- It uses a video compression algorithm based on Discrete Cosine Transforms and Motion Compensation.

Examples:

- AVC[2003]: Advanced Video Coding (Blu-ray, HDTV, Streaming etc.)
- HEVC[2013]: High Efficiency Video Coding (UHD Blu-ray, UHD streaming, macOS High Sierra)
- VVC[2020]: Versatile Video Coding

# Types of Video Compression

There are three main subdivisions in video coding formats:

1. Lossy: Uses partial data and inexact approximations to represent content (mainly found in consumer video).
2. Lossless: Allows the original data to be perfectly reconstructed using the compressed data.
3. Uncompressed: Digital Video that has never been compressed (Clean HDMI, video cameras, video monitors etc.).

# Versatile Video Coding

## Introduction

- Versatile Video Coding (VVC) is a video compression standard finalized on 6 July 2020.
- It is the successor to HEVC.
- The aim is to make 4K broadcast and streaming commercially viable.

## Concept for VVC

- 30–50% better compression rate for the same perceptual quality
- Should support lossless and subjectively lossless compression
- Should support resolutions from 4K to 16K as well as 360° videos

# CABAC

- Contex-based Adaptive Binary Arithmetic Coding is and entropy coding system used in AVC, HEVC and VVC.
- It performs integer bit operations on all floating point type calculations
- CABAC has multiple probability modes for different contexts.
- It first converts all non-binary symbols to binary then, for each bit, the coder selects which probability model to use, and then uses information from nearby elements to optimize the probability estimate.
- Arithmetic coding is finally applied to compress the data.

# CABAC in HEVC

- There are 64 values(finite states) used to represent an accurate estimation.
- Each value has assigned one of 64 representative values dividing the range of $[0.01875, 0.5]$

The update in probabilities in the context model is based on:

$$P(t+1) = \begin{cases} \alpha \cdot P(t) & \text{MPS} \\ 1 - \alpha \cdot (1 - P(t)) & \text{LPS} \end{cases} \tag{1}$$

where, MPS and LPS are the Most Probable Symbols and Least Probable Symbols respectively; $P(t+1)$ is the probability of receiving LPS.

# Determination of $\alpha$

$\alpha$ is the adaptation rate and has a constant value

$$\alpha = \sqrt[63]{0.01875/0.5} \approx 0.949 \tag{2}$$

1. Probability estimation in CABAC is based on a table-driven estimator using a finite-state machine (FSM) approach
2. The method for designing the FSM transition rules was borrowed from HOWARD and VITTER using a model of "exponential aging" which gives rise to the state transition relation (1)

According to (1), the scaling factor a can be determined by

$$P_{min} = 0.5\alpha^{N-1} \tag{3}$$

with the choice of $P_{min} = 0.01875$ and $N = 64$

$$\implies \alpha = \sqrt[63]{0.01875/0.5} \tag{4}$$

# CABAC in VVC

- VVC introduces a new concept that this adaptation rate becomes dynamic during the process of arithmetic coding.
- The context model updates according to the value of the currently encoded symbol.
- The adaptation rate is controlled by two context models in parallel with parameters $r_0$ and $r_1$ :

$$P_i(t+1) = \begin{cases} \left(1 - \frac{1}{2^{r_i}}\right) \cdot P_i(t) & \text{MPS} \\ \\ 1 - \left(1 - \frac{1}{2^{r_i}}\right) \cdot (1 - P_i(t)) & \text{LPS} \end{cases} \tag{5}$$

where $i \in \{0, 1\}$ and $r_i$ is exponentially weighted related to the coding mode.
The updated probability will be the average of $P_i(t+1)$

# CABAC in VVC

1. The purpose of using two parameters is to achieve an optimal update speed.
2. By using different parameters, the shift value can be adapted when the probability changes in the context of obtaining the best effect between the drops rate balances.
3. Instead of the lookup table, VVC uses a linearly quantized probability representation and arithmetic operations to update the probability, which allows adaptation to occur for more situations without the restriction of 64 states in HEVC.

# More Parameter Probability Estimation

In VVC CABAC, there are two context models and their quantization estimator requires 15bits for probability presentation and then 10 and 14 bits for two hypothetical probability estimates:

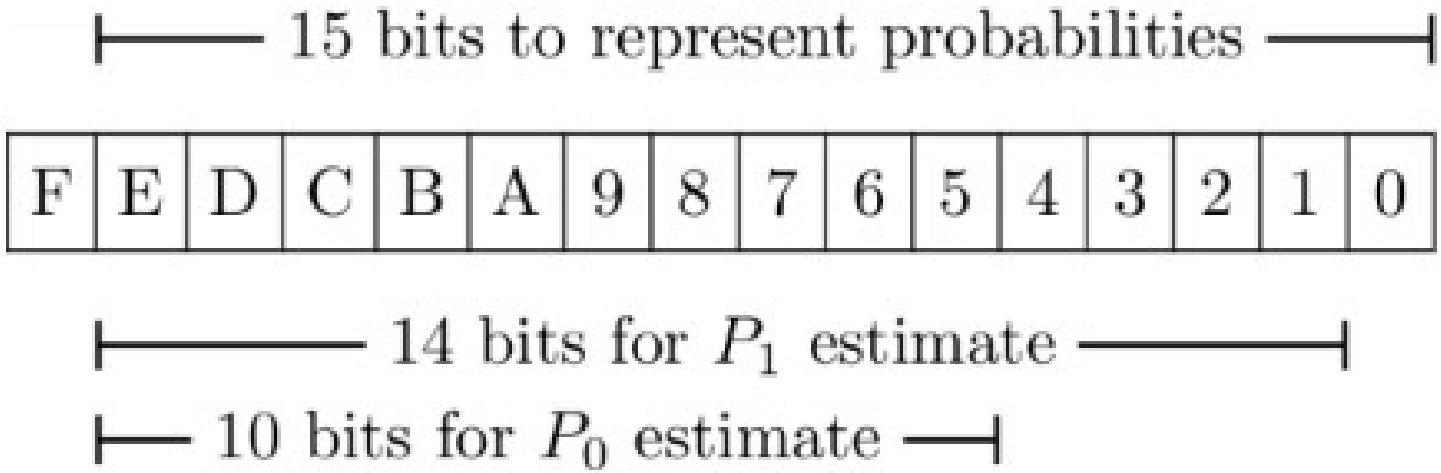


Figure: 15 bits to represent probabilities , 10 and 14 for $P_0$ and $P_1$ resp.

# Continued

For each update of $P_0$ and $P_1$, the high 10 and 14 bits are kept for probability estimation while the low 1 and 5 bits are discarded to 0.

$$0 < P_0 \leq P_1 < 2^{15} \tag{6}$$

While the drop rate $\alpha$ must satisfy

$$\left(1 - \frac{1}{2^0}\right) < \alpha < \left(1 - \frac{1}{2^{15}}\right) \tag{7}$$

Further, since all division is done using bit shift operations, the number of shifts cannot exceed the defined 15 bits.

$$r_i \in \{1, 2 \cdots 14\} \tag{8}$$

$$r_0 < r_1 \tag{9}$$

# Proposed update

Under the new system, the bits are distributed as follows:

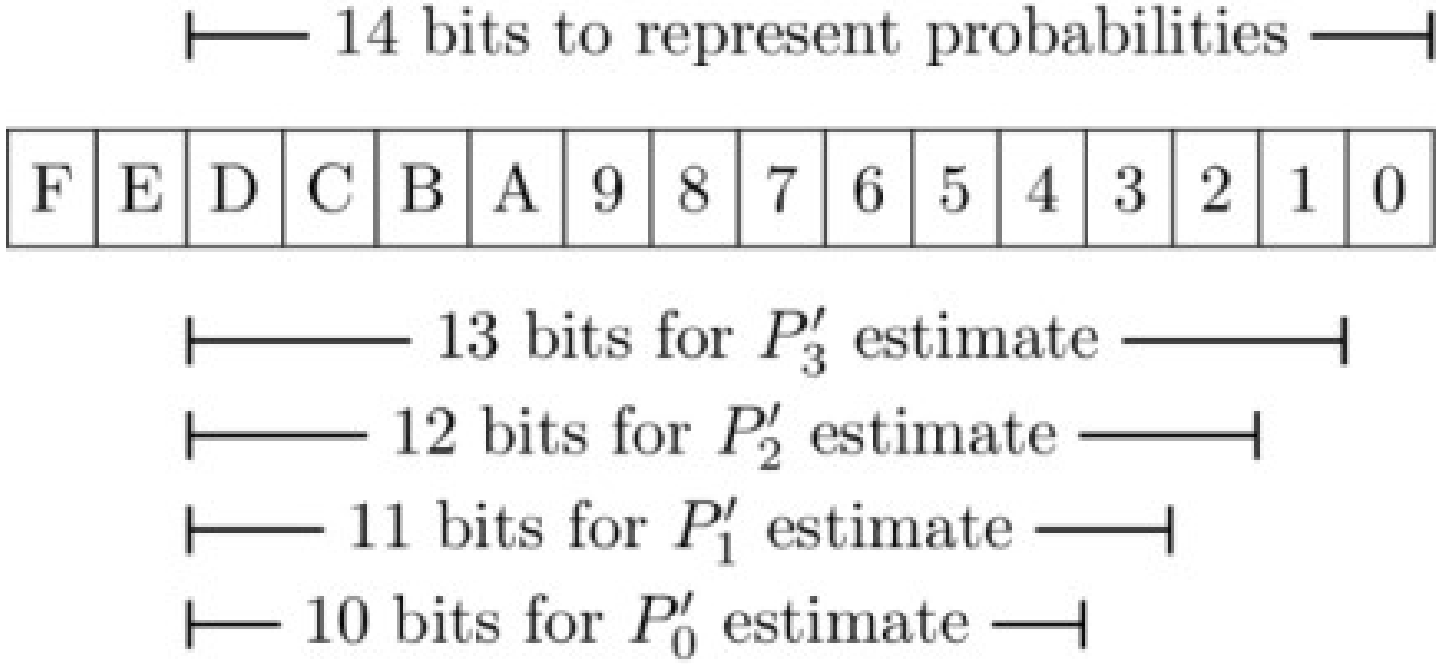


Figure: Proposed set of parameters

- 14 bits to handle probability estimation.
- highest bit to represent MPS symbol
- second highest to prevent overflow.

# Continued

Corresponding to the bit distribution, the following inferences can be made:

$$0 \leq P'_0 \leq P'_1 \leq P'_2 \leq P'_3 < 2^{14} \tag{10}$$

$$\left(1 - \frac{1}{2^0}\right) < \alpha' < \left(1 - \frac{1}{2^{14}}\right) \tag{11}$$

Range of the parameters:

$$r'_i \in \{1, 2, \cdots 13\} \tag{12}$$

$$r'_0 < r'_1 < r'_2 < r'_3 \tag{13}$$

# Drop rate consideration

In the original VVC model, the parameters were preset for different coding modes (so that the drop rates cannot be adjusted at will); the same preset parameters are used in the proposed model:

$$r'_0 = r_0 \tag{14}$$

$$r'_1 = \frac{3}{4} r_0 + \frac{1}{4} r_1 \tag{15}$$

$$r'_2 = \frac{1}{4} r_0 + \frac{3}{4} r_1 \tag{16}$$

$$r'_3 = r_1 \tag{17}$$

(14) and (17) are first used to determine the boundaries of the parameters and the other parameters are calculated from there using linear interpolation.

# Continued

- The proposed model can replace $P_i(t+1)$ in (5) with $P'_i(t+1)$
- Based on the drop rate $r'_i$, four estimators can be determined as follows:

$$P'_i(t+1) = \begin{cases} \left(1 - \frac{1}{2^{r'_i}}\right) \cdot P'_i(t) & \text{MPS} \\ \\ \frac{1}{2^{r'_i}} + \left(1 - \frac{1}{2^{r'_i}}\right) \cdot P'_i(t) & \text{LPS} \end{cases} \tag{18}$$

# MPS Determination

According to the definition of CABAC, $P(t+1)$ is the probability of receiving LPS. This implies $P(t+1) < 0.5$
From (1), we get the following conditions:

$$P(t+1) < P(t) \text{ if MPS is received} \tag{19}$$

$$P(t+1) > P(t) \text{ otherwise} \tag{20}$$

When $P(t+1) > 0.5$, (caused by receiving the LPS continuously) the symbols for MPS and LPS will be swapped. This is used to determine the MPS and it can be applied to the next gen as follows:

$$P'(t+1) = \frac{1}{4}\sum_{i=0}^{3} P'_i(t+1) > 0.5 \tag{21}$$

$$\implies \sum_{i=0}^{3} P'_i(t+1) > 2 \tag{22}$$

# Continued

- The range of probabilities $[0.0, 1.0]$ will be scaled to $[2^0, 2^{14}]$ using integers.
- Thus the decision value in (22) will be scaled to $2 \times 2^{14} = 2^{15}$; at the highest index (F) in Fig. 2
- This is why it was mentioned that the highest bit is reserved for MPS representation.

# Updating process

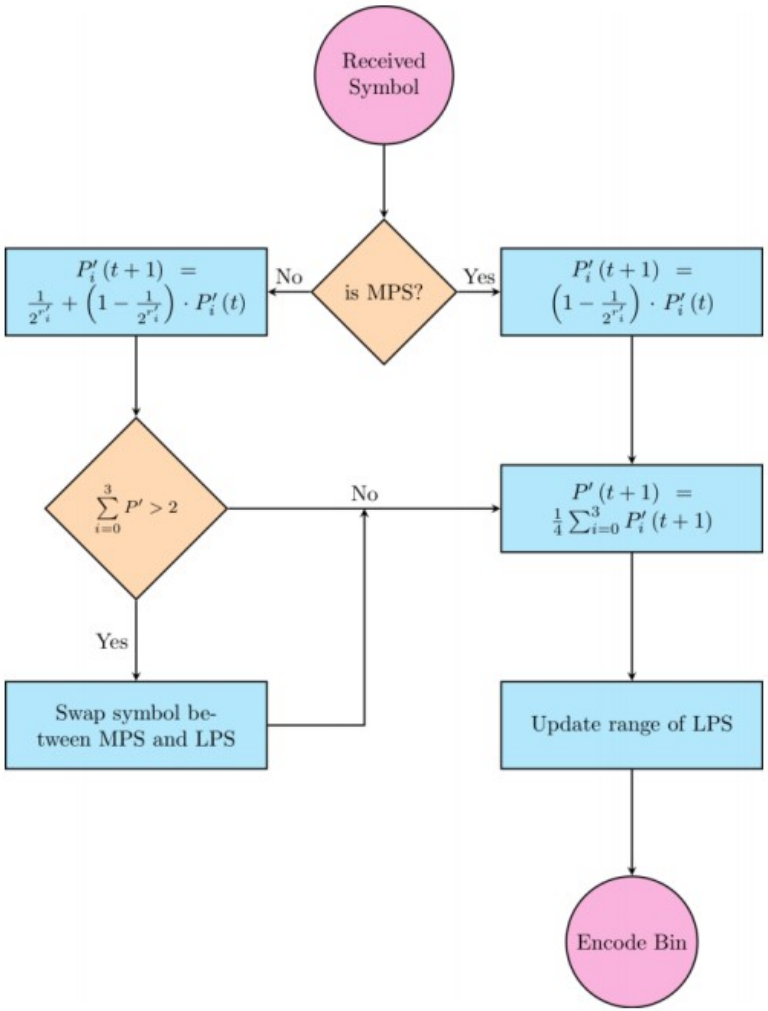


Figure: Flowchart for symbol processing and probability estimation

# Continued

Essence of the flowchart:

1. Check if the received symbol is MPS.
2. if yes, calculate $P_i'(t+1)$ using (5) for MPS.
3. then calculate $P'(t+1)$ by averaging $P_i'(t+1)$
4. accordingly update the range of LPS.
5. if no, calculate $P_i'(t+1)$ using (5) for LPS.
6. check if the sum is greater than 2 (as per (22))
7. if yes, swap MPS and LPS, else simply calculate $P'(t+1)$ and update the range of LPS accordingly.

# Continued

- In HEVC, this part was carried out using a lookup table, so the probabilities can only provide a finite set of values; which is inefficient for floating point calculations.
- Linear quantization is useful as the number of digits in Fig. 2 may increase at any time (for which a lookup table would not work).
- The quantization proposed by VVC requires 10 and 14 bits to store probability estimates of the hypotheses while the method proposed in the paper uses 11-13 bits to achieve better compression efficiency.

# Experimental results

TABLE I. PROPOSED METHODS VS. REFERENCE. APPLYING GOP STRUCTURE AS ALL-INTRA/INTER-IPPPP/INTER-IBBBP

| H.266/VVC | | All-Intra | | Inter-IPPPP | | Inter-IBBBP | |
|---|---|---|---|---|---|---|---|
| Resolution | Sequence | BD-RATE-Y (%) | BD-PSNR-Y (dB) | BD-RATE-Y (%) | BD-PSNR-Y (dB) | BD-RATE-Y (%) | BD-PSNR-Y (dB) |
| Class A 2560 × 1600 | *Traffic* | -2.17 | +0.11 | -0.21 | +0.01 | -0.12 | +0.01 |
| | *PeopleOnStreet* | -1.99 | +0.10 | -0.76 | +0.04 | -0.69 | +0.03 |
| | *NebutaFestival* | -4.27 | +0.30 | -0.50 | +0.04 | -1.50 | +0.04 |
| | *SteamLocomotive* | -3.69 | +0.22 | -2.29 | +0.06 | -2.22 | +0.06 |
| Class B 1920 × 1080 | *Kimono* | -2.46 | +0.10 | -1.00 | +0.04 | -0.83 | +0.03 |
| | *ParkScene* | -2.52 | +0.11 | -0.79 | +0.02 | -0.72 | +0.02 |
| | *Cactus* | -2.23 | +0.08 | -0.80 | +0.02 | -0.70 | +0.02 |
| | *BQTerrace* | -2.26 | +0.11 | -0.96 | +0.02 | -0.79 | +0.02 |
| | *BasketballDrive* | -1.64 | +0.04 | -0.32 | +0.01 | -0.15 | +0.01 |
| Class C 832 × 480 | *RaceHorsesC* | -2.36 | +0.14 | -1.28 | +0.05 | -1.49 | +0.05 |
| | *BQMall* | -1.02 | +0.06 | -1.65 | +0.07 | -2.13 | +0.09 |
| | *PartyScene* | -1.69 | +0.11 | -0.37 | +0.01 | -0.30 | +0.01 |
| Class D 416 × 240 | *RaceHorses* | -0.33 | +0.02 | -3.39 | +0.14 | -4.44 | +0.18 |
| | *BQSquare* | -0.81 | +0.06 | -1.60 | +0.07 | -1.59 | +0.07 |
| | *BlowingBubbles* | -0.12 | +0.01 | -2.26 | +0.08 | -2.28 | +0.08 |
| | *BasketballPass* | -1.42 | +0.08 | -4.56 | +0.20 | -5.00 | +0.22 |
| Class E 1280 × 720 | *FourPeople* | -0.90 | +0.05 | -0.97 | +0.04 | -1.26 | +0.06 |
| | *Johnny* | -0.12 | +0.01 | -2.31 | +0.07 | -2.39 | +0.08 |
| | *KristenAndSara* | -1.01 | +0.05 | -1.57 | +0.06 | -1.45 | +0.05 |
| Class F* | *BasketballDrillText* | -0.48 | +0.02 | -1.02 | +0.05 | -1.45 | +0.06 |
| | *ChinaSpeed* | -1.43 | +0.13 | -0.34 | +0.02 | -0.02 | +0.01 |
| | *SlideEditing* | -1.72 | +0.25 | -1.14 | +0.16 | -1.31 | +0.19 |
| | *SlideShow* | -0.41 | +0.03 | -0.28 | +0.02 | -0.82 | +0.07 |

*Class F is the non-camera captured content such as video screen content.

# Continued

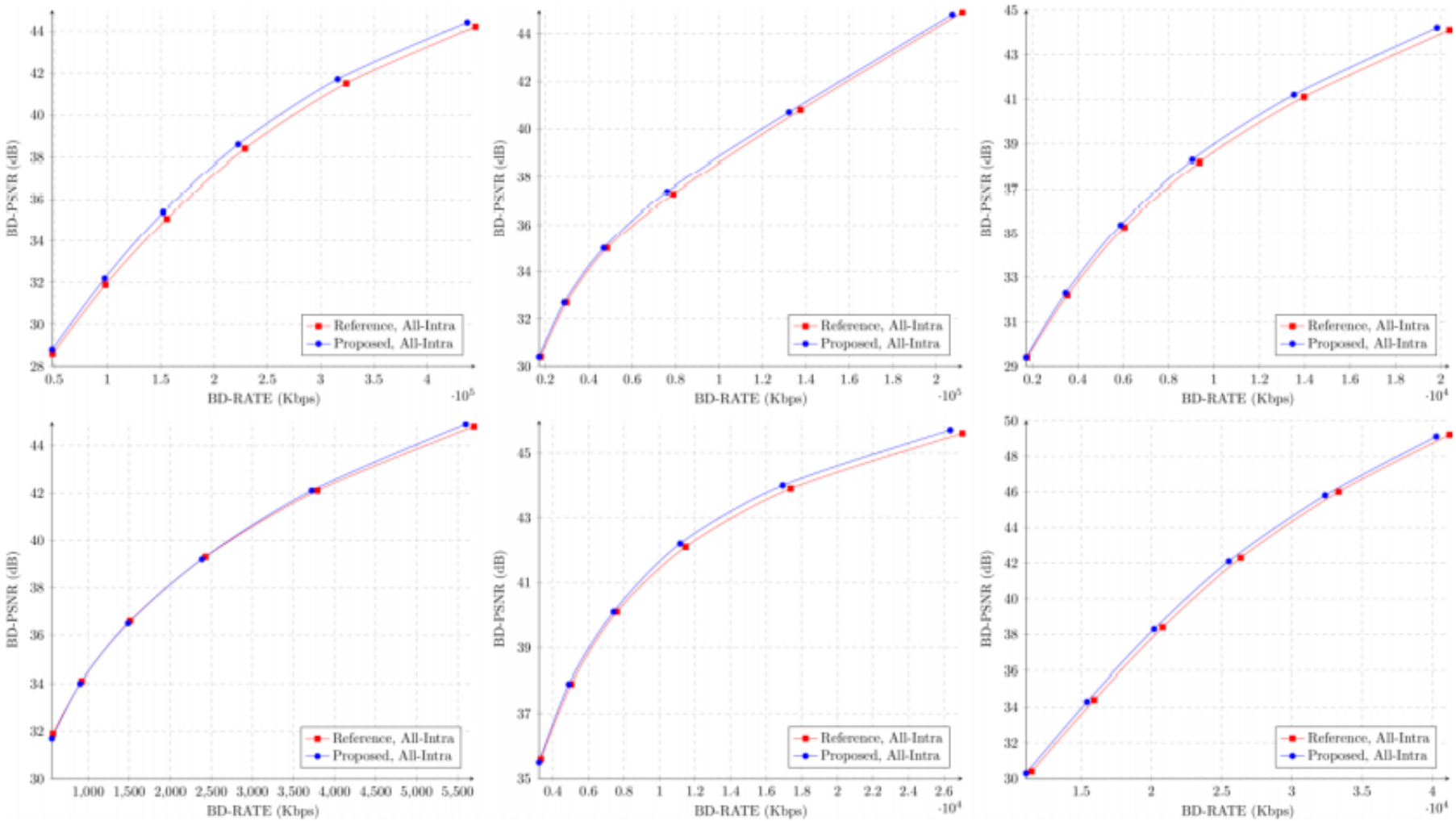


Figure: Graphs for the reference and proposed observations

# In conclusion...

1. Different drop rates are used to achieve flexibility in order to make the estimators usable for sources with different statistical information
2. A major design aspect is the compatibility of having more drop rate decisions and making use of linear quantization instead of finite states in CABAC.
3. This is achieved by using a more compact representation of the probability state, and a multiplication free operation is achieved through a series of bit operations
4. Experiments show that more estimator representations used for encoding lead to better optimization mode decisions without incurring significant complexity overhead.
5. The improvement shows that gain is provided in the encoding, and the proposed method can be used in the next generation VVC standard