STATE-BASED MULTI-PARAMETER PROBABILITY ESTIMATION FOR CONTEXT-BASED ADAPTIVE BINARY ARITHMETIC CODING

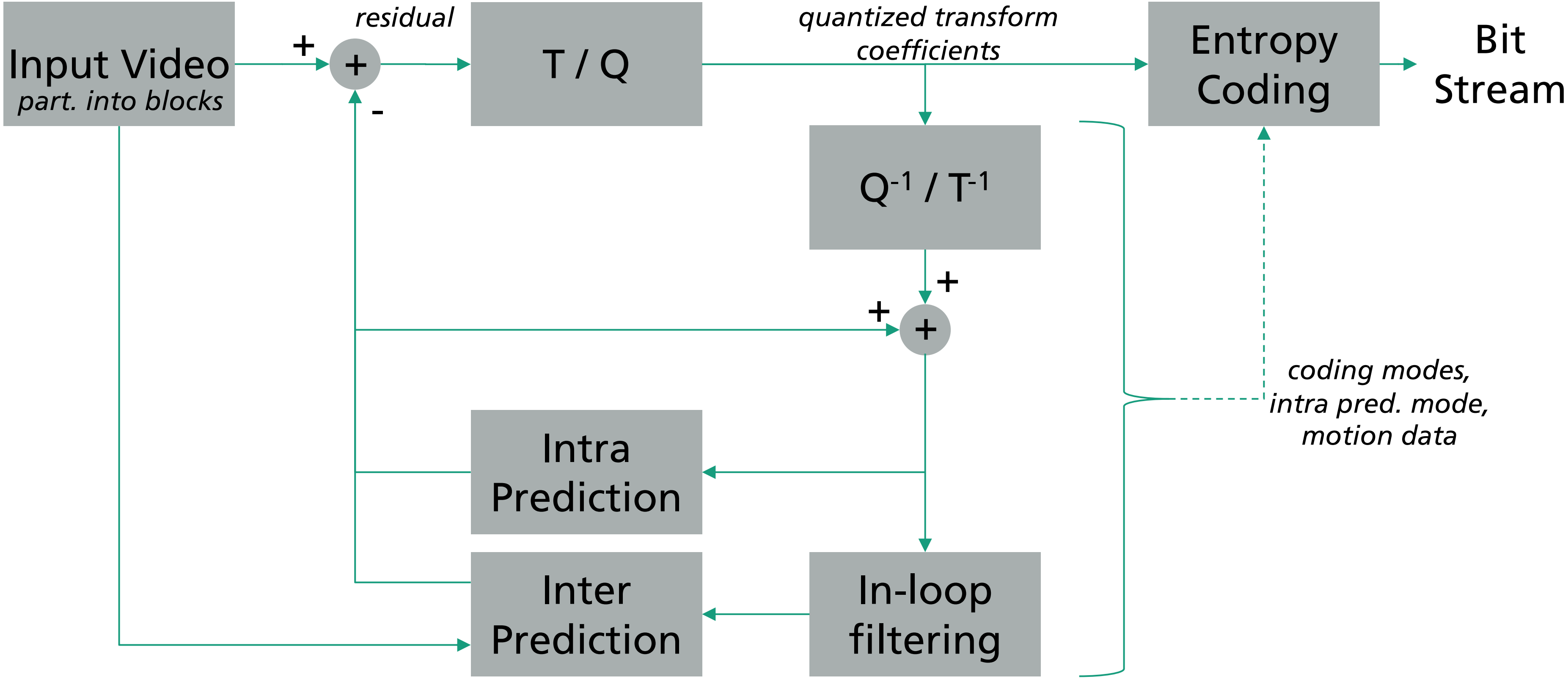
Data Compression Conference 2020

Fraunhofer

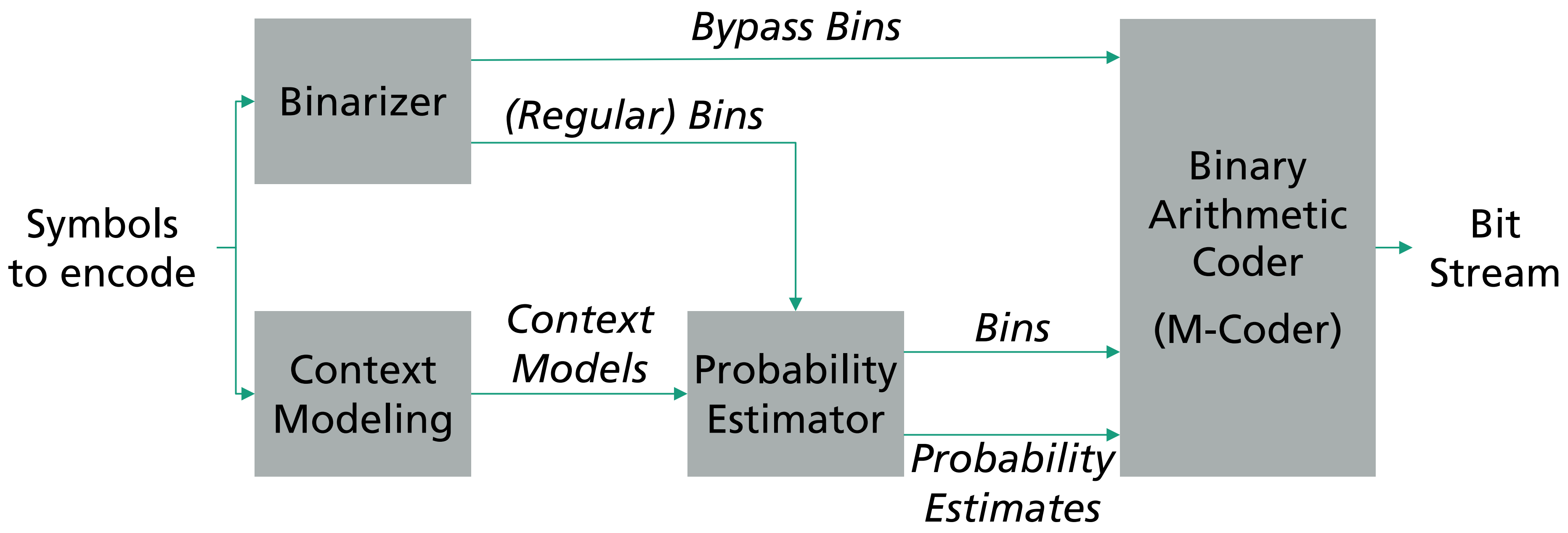
HHI

Paul Haase, Stefan Matlage, Heiner Kirchhoffer, Christian Bartnik, Heiko Schwarz, Detlev Marpe, Thomas Wiegand

Introduction – Hybrid Video Coding Architecture



Introduction – Context-based Adaptive Binary Arithmetic Coding (CABAC)



[Marpe et al., 2003]

Probability Estimation for CABAC

- Compression efficiency largely depends on:
  - Exploiting dependencies (context modeling, probability estimation)
  - Accurate probability estimates
- Sophisticated probability estimation techniques imply higher complexity
  - e.g. memory and computational ressources
- Aim: Enhanced probability estimator
  - High compression efficiency
  - Reduced memory complexity (with respect to comparable approaches)
  - Only additions and bit-shifts

Probability Estimation based on exponentially weighted moving averages (EWMA)

- Probability estimate according to EWMA (or virtual sliding window (VSW)):

$$P(t) = \sigma Y(t) + (1-\sigma)P(t-1)$$

- Impact of previous symbols decreases exponentially with time:

$$P(t) = \sigma(Y(t) + (1-\sigma)Y(t-1) + (1-\sigma)^2 Y(t-2) + \ \ldots + (1-\sigma)^t Y(0))\ , \ \mathrm{WS} = \frac{1}{\sigma}$$

- Quantization of $P(t)$ e.g.:
  - logarithmically (state-based)
  - linearily (counter-based)

$Y(t)$:	time series of binary symbols,
$P(t-1)$:	probability estimate of $Y(t)$ being a ‚1‘
σ:	adaptation rate ($0 \le \sigma \le 1$)

[Belyaev et al., 2006] [Alshin et al., 2013] [Holt, 2004]

Probability Estimation – Logarithmic Quantization of $P(t)$

- LPS/MPS representation (Least / Most probable symbol) introducing:

$$P_{LPS}(t) = 0.5 - |P(t) - 0.5|$$

$$valMPS(t) = \begin{cases} 1, & \text{if } P(t) \geq 0.5 \\ 0, & \text{if } P(t) < 0.5 \end{cases}$$

$$isLPS(t) = Y(t) \oplus valMPS(t-1)$$

- Logarithmic representation and quantization (linear quantization of $S(t)$):

$$P_{LPS}(t) = 0.5 \cdot \alpha^{S(t)}$$

$$R_{LPS}(t) = P_{LPS}(t) \cdot R = 0.5 \cdot \alpha^{S(t)} \cdot R \implies R_{MPS}(t) = R - R_{LPS}(t)$$

[Marpe et al., 2003] [Sullivan et al., 2012]

$\oplus$: exclusive or operator

Multi-Parameter Probability Estimation

- Improvement of the compression efficiency by more accurate probability estimates
- Use of multiple (EWMA) probability estimators with different adaptation rates

$$P_i(t) = \sigma_i Y(t) + (1 - \sigma_i)P_i(t-1), i = \{1, 2, \dots, N\}$$

- Single combined probability estimate by, e.g., (weighted) averaging

$$P_{AVG}(t) = \sum_{i=1}^{N} a_i P_i(t)$$

- E.g. VVC: linearily quantized counter-based approach with two counters (10 an 14 bit)

[Alshin et al., 2013]

SBMP – Logarithmic probability representation

- New signed state variable $U(t)$, incorporating $valMPS(t)$ and $S(t)$:

$$P(t) = \begin{cases} 0.5 \cdot \alpha^{|U(t)|} & \text{, if } U(t) < 0 \\ 1 - 0.5 \cdot \alpha^{|U(t)|} & \text{, if } U(t) \geq 0 \end{cases} \Leftrightarrow P_{LPS}(t) = 0.5 \cdot \alpha^{|U(t)|}$$

- Representation of the MPS value by the sign of $U(t)$:

$$valMPS(t) = \begin{cases} 0, & \text{if } U(t) < 0 \\ 1, & \text{if } U(t) \geq 0 \end{cases}$$

SBMP – Probability updates

- Replacing $P(t)$ with $0.5 \cdot \alpha^{|U(t)|}$ in the probability update function yields two cases depending on the signs of $U(t)$ and $U(t-1)$
- If $U(t)$ and $U(t-1)$ have the same sign:

Case 1: $$|U(t)| = \log_\alpha\left((1-\sigma)\alpha^{|U(t-1)|} + 2\sigma \cdot isLPS(t)\right)$$

- If $U(t)$ and $U(t-1)$ have the opposite sign (only in the LPS case):

Case 2: $$|U(t)| = \log_\alpha\left((\sigma-1)\alpha^{|U(t-1)|} + 2 - 2\sigma\right)$$

- Change of sign means $valMPS(t)$ changes its value

SBMP – Weighted averaging

- $U_i(t)$: State variable $U(t)$ of estimator i of a multi-parameter probability estimator
- Linear averaging requires conversion from $U_i(t)$ to $P_i(t)$ for all i
- Alternative: directly averaging all $U_i(t)$:

$$U_{AVG}(t) = \sum_{i=1}^{N} b_i U_i(t)$$

- When all $U_i(t)$ have the same sign, this corresponds to weighted geometric average. E.g. for all $U_i(t) < 0$ and all $b_i = {}^1/_N$:

$$P_{GEO}(t) = \sqrt[N]{\prod_{i=1}^{N} P_i(t)} = \sqrt[N]{0.5^N \cdot \alpha^{\sum_{i=1}^{N} |U_i(t)|}} = 0.5 \cdot \alpha^{\frac{1}{N}\sum_{i=1}^{N} |U_i(t)|}$$

SBMP – State quantization and choice of α

- Logarithmic quantization of $P_i(t) \Leftrightarrow$ linear quantization of $U_i(t)$
- Choice of α, so that all σ can be represented $\Longrightarrow$ Change of $U_i(t)$ by at least 1
- Case 1:

$$|U(t) - U(t-1)| = \big||U(t)| - |U(t-1)|\big| = \left|\log_\alpha\left(1 - \sigma + \frac{2\sigma \cdot isLPS(t)}{\alpha^{|U(t-1)|}}\right)\right|$$

- Case 2:

$$|U(t) - U(t-1)| = |U(t)| + |U(t-1)| = \log_\alpha\left((1-\sigma)\left(2\alpha^{|U(t-1)|} - \alpha^{2|U(t-1)|}\right)\right)$$

$$\Longrightarrow \alpha = 1 - \sigma$$

SBMP – Exemplary configuration

- Quantize $U_1(t)$ and $U_2(t)$ to signed integers with 8 bit and 12 bit
- Smallest chosen adaptation rate $\sigma_2 = {}^{1}/_{944.1} \Leftrightarrow \alpha_2 \approx 0.99894079$
- Set $\alpha_1 = {\alpha_2}^{16} \Leftrightarrow \sigma_1 \approx 59.476 \Longrightarrow$ simple conversions between $U_1(t)$ and $U_2(t)$

$$P_{LPS}(t) = 0.5 \cdot \left(1 - \frac{1}{944.1}\right)^{|16 \cdot U_1(t) + U_2(t)|}$$

- Interval subdivision using a look-up table with 256 entries:

$$R_{LPS}(t) = LT_{LPS}[|U_2(t) + (U_1(t) \ll 4)| \gg 7][(R \gg 5)\ \&7]$$

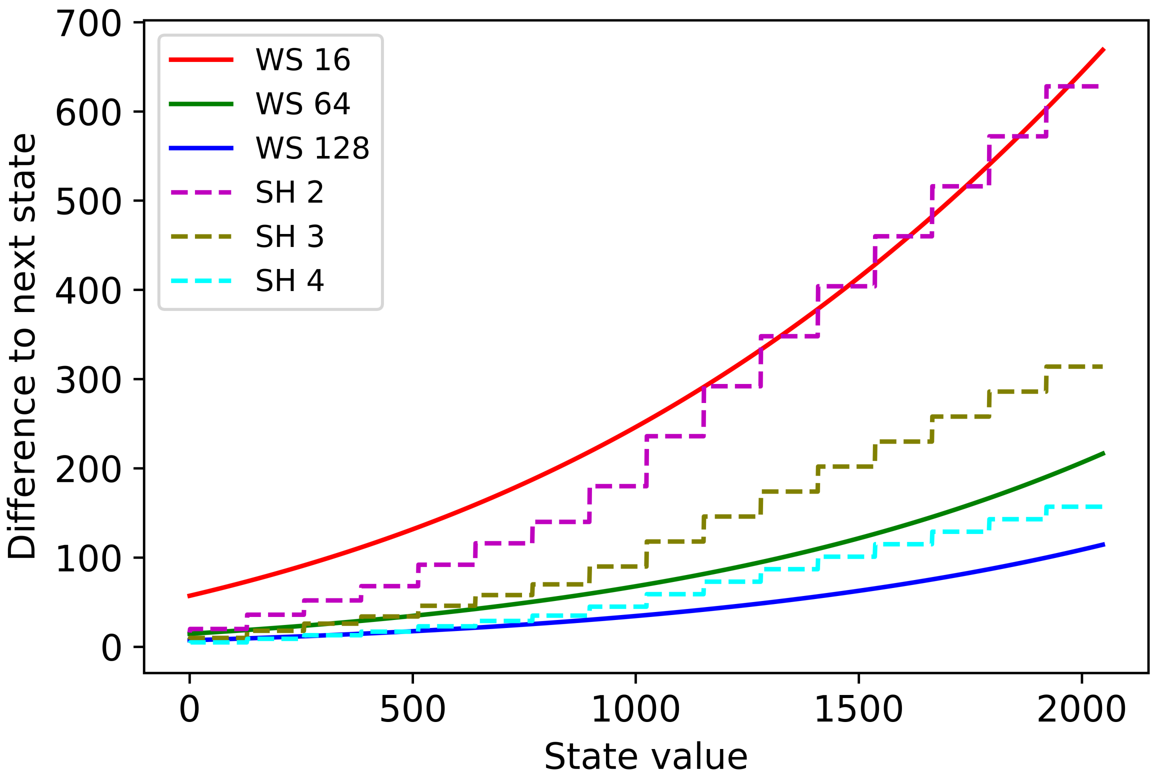
SBMP – State transition and update process

- Update of two states $U_1(t)$, $U_2(t)$ (with different adaptation rates σ_1, σ_2) using only one look-up table
  - Containing the differences between $U(t)$ and $U(t-1) \Rightarrow$ Table size reduction to only 32 elements

$$U_i(t) = U_i(t-1) \pm (LT_U[16 + (\pm U_i(t-1) \gg s_i)] \gg sh_i)$$

- Shifts sh_i instead of different adaptation rates σ_i
  - Similar behaviour, but only one LUT required

SBMP – State transition and update process



Update Look-Up Table LT_U			
0	2512	11	368
1	2288	12	272
2	2064	13	208
3	1840	14	144
4	1616	15	80
5	1392	16	64
6	1168	17	64
7	944	…	…
8	720	29	64
9	560	30	64
10	464	31	0

SBMP – Experimental evaluation – VTM-2.0.1

- Reference: VTM-2.0.1
  - Test Model of new standardization activity Versatile Video Coding
  - Similar Design as HEVC
- Common Test Conditions of Joint Video Experts Team (JVET CT)
  - Test scenarios
    - all intra
    - random access
    - low delay (not used)
  - Base QPs: 37, 32, 27, 22
  - Bit-rate savings in terms of Bjøntegaard delta rate

[Bossen et al., 2018]

[Bjøntegaard, 2001]

SBMP – Experimental evaluation – VTM-2.0.1

- JVET Test Sequences
  - Class A1 – 3 Sequences – 3840 x 2160
  - Class A2 – 3 Sequences – 3840 x 2160
  - Class B – 5 Sequences – 1920 x 1080
  - Class C – 4 Sequences – 832 x 480
  - Class D – 4 Sequences – 416 x 240 – not included in average
  - Class E – 3 Sequences – 1280 x 720
  - Class F – 4 Sequences – 832 x 480 – 1920 x 1080 – screen content – not included in average

[Bossen et al., 2018]

SBMP – Experimental evaluation – VTM-2.0.1

Seq. Class	SBMP						Counter-based (10/14 Bit, similar to VVC)					
	YUV-BD-Rates (AI)			YUV-BD-Rates (RA)			YUV-BD-Rates (AI)			YUV-BD-Rates (RA)		
	Y (%)	U (%)	V (%)	Y (%)	U (%)	V (%)	Y (%)	U (%)	V (%)	Y (%)	U (%)	V (%)
A1	-1.18	-1.28	-1.63	-0.95	-0.77	-1.30	-1.15	-1.23	-1.08	-0.99	-1.07	-0.97
A2	-0.67	-1.52	-1.31	-0.64	-0.40	-0.72	-0.97	-1.39	-1.26	-0.84	-0.71	-0.79
B	-1.03	-1.80	-1.77	-0.92	-1.12	-0.80	-1.03	-1.09	-1.20	-1.00	-1.22	-0.72
C	-0.99	-1.71	-1.63	-0.87	-1.29	-1.05	-0.97	-1.13	-1.05	-0.90	-1.07	-0.96
E	-0.89	-1.55	-1.64				-0.91	-1.15	-1.45			
Avg	**-0.96**	**-1.61**	**-1.62**	**-0.86**	**-0.95**	**-0.95**	**-1.01**	**-1.18**	**-1.20**	**-0.94**	**-1.05**	**-0.85**
D	-0.88	-1.18	-1.63	-0.82	-1.41	-1.16	-0.86	-0.80	-1.28	-0.80	-1.39	-0.90
F	-0.79	-1.26	-1.24	-0.67	-0.79	-0.88	-0.83	-1.06	-1.02	-0.78	-0.89	-0.88

Conclusion

- Adavanced state-based probability estimator
  - -0.96% (AI) and -0.86 (RA) luma BD-rate gain (compared to HEVC scheme)
- Minor loss but lower memory complexity compared to similar counter-based approaches
  - Loss of 0.04% (AI) and 0.08% (RA) with respect to a counter-based approach almost identical to VVC
  - 8/12 Bit states instead of 10/14 Bit counters saving:
    - 4 bits per context model
    - 4000 Bits in VTM-2.0.1 (approx. 1000 context models) at a cost of 2304 Bits for the look-up tables
- No multiplications or divisions (only additions and table look-ups)

THANK YOU!