Claims

1. A method for decoding a video, the method comprising:

accessing a binary string representing a partition of the video, the partition comprising a plurality of coding tree units (CTUs) forming one or more CTU rows;

for each CTU of the plurality of CTUs in the partition,

prior to decoding the CTU, determining that the CTU is a first CTU of a current CTU row among the one or more CTU rows in the partition;

in response to determining that the CTU is the first CTU of the current CTU row in the partition,

setting a history counter for a color component for calculating Rice parameters to an initial value;

decoding the CTU, comprising:

calculating the Rice parameters for transform units (TUs) in the CTU based on the history counter;

decoding the binary string corresponding to the TUs in the CTU into coefficient values of the TUs based on the calculated Rice parameters; and

determining pixel values for the TUs in the CTU from the coefficient values; and

outputting a decoded partition of the video comprising the decoded plurality of CTUs in the partition,

wherein setting the history counter for the color component cIdx for the calculating Rice parameters to the initial value comprises:

in response to determining that a history-based Rice parameter derivation is enabled, calculating the initial value by

StatCoeff [cIdx] = 2 * Floor(Log2(BitDepth − 10)),

wherein StatCoeff denotes the history counter, BitDepth specifies a bit depth of samples of luma and chroma arrays of the video, Floor(x) represents the largest integer less than or equal to x, and Log2(x) is base-2 logarithm of x.

2. The method of claim 1, wherein the partition is a frame, or a slice, or a tile.

3. The method of claim 1, wherein calculating the Rice parameters for the TUs in the CTU based on the history counter comprises:

determining a replacement variable HistValue based on the history counter;

calculating a local sum variable locSumAbs for a coefficient in a TU of the CTU using values of neighboring coefficients in a pre-determined neighborhood of the coefficient and the replacement variable HistValue; and

deriving the Rice parameter for the TU based on the local sum variable locSumAbs.

4. The method of claim 3, wherein determining the replacement variable HistValue based on the history counter comprises determining the replacement variable HistValue for the color component cIdx by calculating:

$$HistValue[cIdx] = 1 << StatCoeff[cIdx],$$

wherein StatCoeff denotes the history counter.

5. The method of claim 3, wherein calculating the local sum variable locSumAbs for the coefficient in the TU of the CTU comprises:

determining that a neighboring coefficient of a plurality of neighboring coefficients in the pre-determined neighborhood of the coefficient is outside the TU; and

using the replacement variable HistValue as a value of the neighboring coefficient outside the TU to calculate the local sum variable locSumAbs.

6. The method of claim 1, wherein decoding the CTU further comprises updating the history counter by:

in response to determining that a first, non-zero, Golomb-Rice coded transform coefficient in a TU is coded as abs_remainder, updating the history counter for the color component cIdx as:

$StatCoeff[cIdx] = (StatCoeff[cIdx] + Floor(Log2(abs_remainder[cIdx])) + 2) >> 1$; and

in response to determining that the first, non-zero, Golomb-Rice coded transform coefficient in the TU is coded as dec_abs_level, updating the history counter for the color component cIdx as:

StatCoeff[cIdx] = (StatCoeff[cIdx] + Floor(Log2(dec_abs_level[cIdx]))) >> 1, wherein StatCoeff denotes the history counter, Floor(x) represents the largest integer less than or equal to x, and Log2(x) is a base-2 logarithm of x.

7. A non-transitory computer-readable medium having program code that is stored thereon, the program code executable by one or more processing devices for performing the method of any one of claims 1 to 6.

8. A system comprising:

a processing device; and

a non-transitory computer-readable medium communicatively coupled to the processing device, wherein the processing device is configured to execute program code stored in the non-transitory computer-readable medium and thereby perform the method of any one of claims 1 to 6.

9. A method for encoding a video, the method comprising:

accessing a partition of the video, the partition comprising a plurality of coding tree units (CTUs) forming one or more CTU rows;

processing the partition of the video to generate a binary representation of the partition, the processing comprising:

for each CTU of the plurality of CTUs in the partition,

prior to encoding the CTU, determining that parallel coding is enabled and that the CTU is a first CTU of a current CTU row among the one or more CTU rows in the partition;

in response to determining that the parallel coding is enabled and that the CTU is the first CTU of the current CTU row in the partition, setting a history counter for a color component for calculating Rice parameters to an initial value; and

encoding the CTU, comprising:

calculating the Rice parameters for transform units (TUs) in the CTU based on the history counter, and

encoding coefficient values of the TUs into a binary representation corresponding to the TUs in the CTU based on the calculated Rice parameters; and

encoding the binary representation of the partition into a bitstream of the video,

wherein setting the history counter for the color component cIdx for calculating the Rice parameters to the initial value comprises:

in response to determining that a history-based Rice parameter derivation is enabled, calculating the initial value by

StatCoeff [cIdx] = 2 * Floor(Log2(BitDepth − 10)),

wherein StatCoeff denotes the history counter, BitDepth specifies a bit depth of samples of luma and chroma arrays of the video, Floor(x) represents the largest integer less than or equal to x, and Log2(x) is base-2 logarithm of x.

10. The method of claim 9, wherein the partition is a frame, or a slice, or a tile.

11. The method of claim 9, wherein calculating the Rice parameters for the TUs in the CTU based on the history counter comprises:

determining a replacement variable HistValue based on the history counter;

calculating a local sum variable locSumAbs for a coefficient in a TU of the CTU using values of neighboring coefficients in a pre-determined neighborhood of the coefficient and the replacement variable HistValue; and

deriving the Rice parameter for the TU based on the local sum variable locSumAbs.

12. The method of claim 11, wherein determining the replacement variable HistValue based on the history counter comprises determining the replacement variable HistValue for the color component cIdx by calculating:

HistValue [cIdx] = 1 << StatCoeff[cIdx],

wherein StatCoeff denotes the history counter.

13. The method of claim 11, wherein calculating the local sum variable locSumAbs for the coefficient in the TU of the CTU comprises:

determining that a neighboring coefficient of a plurality of neighboring coefficients in the pre-determined neighborhood of the coefficient is outside the TU; and

using the replacement variable HistValue as a value of the neighboring coefficient outside the TU to calculate the local sum variable locSumAbs.

14. The method of claim 9, wherein encoding the CTU further comprises updating the history counter by:

in response to determining that a first, non-zero, Golomb-Rice coded transform coefficient in a TU is coded as abs_remainder, updating the history counter for the color component cIdx as:

StatCoeff[cIdx] = (StatCoeff[cIdx] + Floor(Log2(abs_remainder[cIdx])) + 2) >> 1; and

in response to determining that the first, non-zero, Golomb-Rice coded transform coefficient in the TU is coded as dec_abs_level, updating the history counter for the color component cIdx as:

StatCoeff[cIdx] = (StatCoeff[cIdx] + Floor(Log2(dec_abs_level[cIdx]))) >> 1, wherein StatCoeff denotes the history counter, Floor(x) represents the largest integer less than or equal to x, and Log2(x) is a base-2 logarithm of x.

15. A non-transitory computer-readable medium having program code and a bitstream that is stored thereon, the program code, when executed by one or more processing devices, causes the one or more processing devices to perform the method of any one of claims 9 to 14 to generate the bitstream.

16. A system comprising:

a processing device; and

a non-transitory computer-readable medium communicatively coupled to the processing device, wherein the processing device is configured to execute program code stored in the non-transitory computer-readable medium and thereby perform the method of any one of claims 9 to 14.