(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
04 June 2020 (04.06.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/108591 A1

(51) International Patent Classification:
H04N 19/117 (2014.01) *H04N 19/159* (2014.01)

(21) International Application Number:
PCT/CN2019/121850

(22) International Filing Date:
29 November 2019 (29.11.2019)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PCT/CN2018/118799
01 December 2018 (01.12.2018) CN
PCT/CN2018/119709
07 December 2018 (07.12.2018) CN
PCT/CN2018/125412
29 December 2018 (29.12.2018) CN
PCT/CN2019/070002
01 January 2019 (01.01.2019) CN
PCT/CN2019/075874
22 February 2019 (22.02.2019) CN
PCT/CN2019/075993
24 February 2019 (24.02.2019) CN
PCT/CN2019/076195
26 February 2019 (26.02.2019) CN
PCT/CN2019/079396
24 March 2019 (24.03.2019) CN
PCT/CN2019/079431
25 March 2019 (25.03.2019) CN
PCT/CN2019/079769
26 March 2019 (26.03.2019) CN

(71) Applicants: **BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD.** [CN/CN]; Room B-0035, 2/F, No.3 Building, No.30, Shixing Road, Shijingshan District, Beijing 100041 (CN). **BYTEDANCE INC.** [US/US]; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US).

(72) Inventors: **ZHANG, Kai**; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). **ZHANG, Li**; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). **LIU, Hongbin**; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road, Haidian District, Beijing 100080 (CN). **XU, Jizheng**; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). **WANG, Yue**; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road, Haidian District, Beijing 100080 (CN).

(74) Agent: **LIU, SHEN & ASSOCIATES**; 10th Floor, Building 1, 10 Caihefang Road, Haidian District, Beijing 100080 (CN).

(54) Title: PARAMETER DERIVATION FOR INTRA PREDICTION

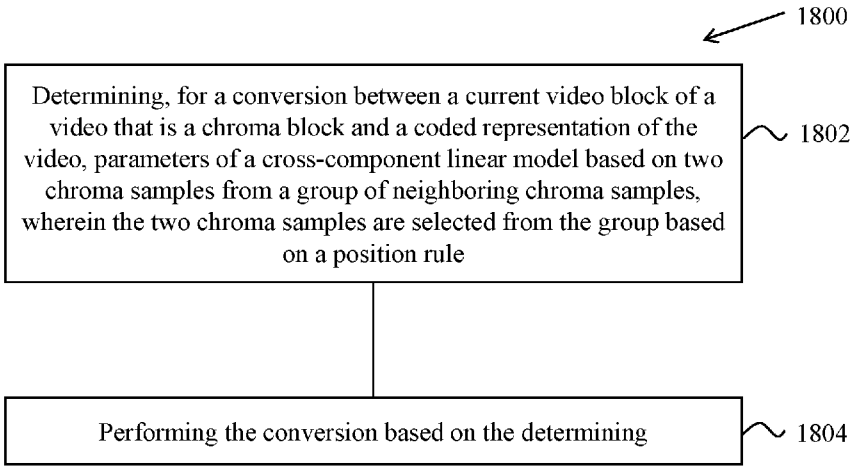

FIG. 18

(57) Abstract: A method for video processing is provided. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode; and performing the conversion based on the determining.

WO 2020/108591 A1

[Continued on next page]

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

— *with international search report (Art. 21(3))*

PARAMETER DERIVATION FOR INTRA PREDICTION

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] Under the applicable patent law and/or rules pursuant to the Paris Convention, this application is made to timely claim the priority to and benefits of International Patent Application No. PCT/CN2018/118799, filed on December 1, 2018, International Patent Application No. PCT/CN2018/119709, filed on December 7, 2018, International Patent Application No. PCT/CN2018/125412, filed on December 29, 2018, International Patent Application No. PCT/CN2019/070002, filed on January 1, 2019, International Patent Application No. PCT/CN2019/075874, filed on February 22, 2019, International Patent Application No. PCT/CN2019/075993, filed on February 24, 2019, International Patent Application No. PCT/CN2019/076195, filed on February 26, 2019, International Patent Application No. PCT/CN2019/079396, filed on March 24, 2019, International Patent Application No. PCT/CN2019/079431, filed on March 25, 2019, and International Patent Application No. PCT/CN2019/079769, filed on March 26, 2019. For all purposes under the U.S. law, the entire disclosures of the aforementioned applications are incorporated by reference as part of the disclosure of this application.

TECHNICAL FIELD

[0002] This patent document relates to video processing techniques, devices and systems.

BACKGROUND

[0003] In spite of the advances in video compression, digital video still accounts for the largest bandwidth use on the internet and other digital communication networks. As the number of connected user devices capable of receiving and displaying video increases, it is expected that the bandwidth demand for digital video usage will continue to grow.

SUMMARY

[0004] Devices, systems and methods related to digital video processing, and for example, simplified linear model derivations for the cross-component linear model (CCLM) prediction

mode in video coding are described. The described methods may be applied to both the existing video coding standards (e.g., High Efficiency Video Coding (HEVC)) and future video coding standards (e.g., Versatile Video Coding (VVC)) or codecs.

[0005] In one representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on two chroma samples from a group of neighboring chroma samples, wherein the two chroma samples are selected from the group based on a position rule; and performing the conversion based on the determining.

[0006] In one representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on selected chroma samples based on positions of the chroma samples, wherein the selected chroma samples are selected from a group of neighboring chroma samples, and performing the conversion based on the determining.

[0007] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a current video block, a group of neighboring chroma samples used to derive a set of values for parameters of a linear model, wherein a width and a height of the current video block is W and H, respectively, and wherein the group of neighboring chroma samples comprises at least one sample that is located beyond $2\times W$ above neighboring chroma samples or $2\times H$ left neighboring chroma samples; and performing, based on the linear model, a conversion between the current video block and a coded representation of a video including the current video block.

[0008] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, multiple sets of parameters, wherein each set of parameters defines a cross-component linear model (CCLM) and is derived from a corresponding group of chroma samples at corresponding chroma sample positions; determining, based on the multiple sets of parameters, parameters for a final CCLM; and performing the conversion based on the final CCLM.

[0009] In another representative aspect, the disclosed technology may be used to provide a

method for video processing. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on maximum and minimum values of chroma and luma samples of N groups of chroma and luma samples selected from neighboring luma and chroma samples of the current video block; and performing the conversion using the CCLM.

[0010] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model that are completely determinable by two chroma samples and corresponding two luma samples; and performing the conversion based on the determining.

[0011] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model using a parameter table whose entries are retrieved according to two chroma sample values and two luma sample values; and performing the conversion based on the determining.

[0012] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a final prediction P(x, y) of a chroma sample at a position (x, y) in the current video block as a combination of prediction results of multiple cross-component linear models (MCCLMs), wherein the MCCLMs are selected based on the position (x, y) of the chroma sample; and performing the conversion based on the final prediction.

[0013] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes performing, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a first determination regarding whether a first cross-component linear model (CCLM) that uses only left-neighboring samples is used for predicting samples of the current video block and/or a second determination regarding whether a second cross-component linear model (CCLM) that uses only above-neighboring samples is used for predicting samples of the current video block; and

performing the conversion based on the first determination and/or the second determination.

[0014] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode; and performing the conversion based on the determining, wherein the flag is signaled to indicate whether the CCLM prediction mode is applied on the current video block, and wherein the CCLM prediction mode uses a linear mode to derive prediction values of a chroma component from another component.

[0015] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, a coding order for one or more indications of a derived mode (DM mode) and a linear mode (LM mode) based on a coding mode of one or more neighboring blocks of the current video block; and performing the conversion based on the determining, wherein the LE mode uses a linear mode to derive prediction values of a chroma component from another component, and the DM mode derives intra prediction mode of a chroma component from another component.

[0016] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on refined neighboring luma samples and chroma samples of the current video block; deriving prediction values of a chroma component of the current video block based on the parameters and refined internal luma samples of the current video block; and performing the conversion based on the prediction values.

[0017] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on by selecting neighboring samples based on a position of a largest or a smallest neighboring sample; deriving prediction values of chroma samples of the current video block based on the parameters and

internal luma samples of the current video block; and performing the conversion based on the prediction values.

[0018] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method includes determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on a main color component and a dependent color component, the main color component selected as one of a luma color component and a chroma color component and the dependent color component selected as the other of the luma color component and the chroma color component; and performing the conversion based on the determining.

[0019] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: performing downsampling on chroma and luma samples of a neighboring block of the current video block; determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on the downsampled chroma and luma samples obtained from the downsampling; and performing the conversion based on the determining.

[0020] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or more chroma samples from a group of neighboring chroma samples, wherein the two or more chroma samples are selected based on a coding mode of the current video block; and performing the conversion based on the determining.

[0021] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on W available above-neighboring samples, W being an integer; and performing the conversion based on the determining.

[0022] In another representative aspect, the disclosed technology may be used to provide a

method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on H available left-neighboring samples of the current video block; and performing the conversion based on the determining.

[0023] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or four chroma samples and/or corresponding luma samples; and performing the conversion based on the determining.

[0024] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: selecting, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, chroma samples based on a position rule, the chroma samples used to derive parameters of a cross-component linear model (CCLM); and performing the conversion based on the determining, wherein the position rule specifies to select the chroma samples that are located within an above row and/or a left column of the current video block.

[0025] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, positions at which luma samples are downsampled, wherein the downsampled luma samples are used to determine parameters of a cross-component linear model (CCLM) based on chroma samples and downsampled luma samples, wherein the downsampled luma samples are at positions corresponding to positions of the chroma samples that are used to derive the parameters of the CCLM; and performing the conversion based on the determining.

[0026] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a method to derive parameters of a cross-component linear model (CCLM) using chroma samples and luma samples based on a coding condition associated with the current video block; and performing the conversion based on the determining.

[0027] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, whether to derive maximum values and/or minimum values of a luma component and a chroma component that are used to derive parameters of a cross-component linear model (CCLM) based on availability of a left-neighboring block and an above-neighboring block of the current video block; and performing the conversion based on the determining.

[0028] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a coding tool using a linear model based on selected neighboring samples of the current video block and corresponding neighboring samples of a reference block; and performing the conversion based on the determining.

[0029] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a local illumination compensation (LIC) tool based on N neighboring samples of the current video block and N corresponding neighboring samples of a reference block, wherein the N neighboring samples of the current video block are selected based on positions of the N neighboring samples; and performing the conversion based on the determining, wherein the LIC tool uses a linear model of illumination changes in the current video block during the conversion.

[0030] In another representative aspect, the disclosed technology may be used to provide a method for video processing. The method comprises determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on chroma samples and corresponding luma samples; and performing the conversion based on the determining, wherein some of the chroma samples are obtained by a padding operation and the chroma samples and the corresponding luma samples are grouped into two arrays G0 and G1, each array including two chroma samples and corresponding luma samples.

[0031] In yet another representative aspect, the above-described method is embodied in the form of processor-executable code and stored in a computer-readable program medium.

[0032] In yet another representative aspect, a device that is configured or operable to perform the above-described method is disclosed. The device may include a processor that is programmed to implement this method.

[0033] In yet another representative aspect, a video decoder apparatus may implement a method as described herein.

[0034] The above and other aspects and features of the disclosed technology are described in greater detail in the drawings, the description and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0035] FIG. 1 shows an example of locations of samples used for the derivation of the weights of the linear model used for cross-component prediction.

[0036] FIG. 2 shows an example of classifying neighboring samples into two groups.

[0037] FIG. 3A shows an example of a chroma sample and its corresponding luma samples.

[0038] FIG. 3B shows an example of down filtering for the cross-component linear model (CCLM) in the Joint Exploration Model (JEM).

[0039] FIGS. 4A and 4B show examples of only top-neighboring and only left-neighboring samples used for prediction based on a linear model, respectively.

[0040] FIG. 5 shows an example of a straight line between minimum and maximum luma values as a function of the corresponding chroma samples.

[0041] FIG. 6 shows an example of a current chroma block and its neighboring samples.

[0042] FIG. 7 shows an example of different parts of a chroma block predicted by a linear model using only left-neighboring samples (LM-L) and a linear model using only above-neighboring samples (LM-A).

[0043] FIG. 8 shows an example of a top-left neighboring block.

[0044] FIG. 9 shows an example of samples to be used to derive a linear model.

[0045] FIG. 10 shows an example of left and below-left columns and above and above-right rows relative to a current block.

[0046] FIG. 11 shows an example of a current block and its reference samples.

[0047] FIG. 12 shows examples of two neighboring samples when both left and above neighboring reference samples are available.

[0048] FIG. 13 shows examples of two neighboring samples when only above neighboring

reference samples are available.

[0049] FIG. 14 shows examples of two neighboring samples when only left neighboring reference samples are available.

[0050] FIG. 15 shows examples of four neighboring samples when both left and above neighboring reference samples are available.

[0051] FIG. 16 shows an example of lookup tables used in LM derivations.

[0052] FIG. 17 shows an example of an LM parameter derivation process with 64 entries.

[0053] FIG. 18 shows a flowchart of an example method for video processing based on some implementations of the disclosed technology.

[0054] FIGS. 19A and 19B show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0055] FIG. 20A and 20B show flowcharts of another example methods for video processing based on some implementations of the disclosed technology.

[0056] FIG. 21 shows a flowchart of another example method for video processing based on some implementations of the disclosed technology.

[0057] FIG. 22 shows a flowchart of an example method for video processing based on some implementations of the disclosed technology.

[0058] FIGS. 23A and 23B show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0059] FIGS. 24A-24E show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0060] FIGS. 25A and 25B show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0061] FIGS. 26A and 26B show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0062] FIGS. 27A and 27B show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0063] FIGS. 28A-28C show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0064] FIGS. 29A-29C show flowcharts of example methods for video processing based on some implementations of the disclosed technology.

[0065] FIGS. 30A and 30B are block diagrams of examples of hardware platforms for implementing a visual media decoding or a visual media encoding technique described in the present document.

[0066] FIGS. 31A and 31B show examples of LM parameter derivation process with four entries. FIG. 31A shows an example when both above and left neighboring samples are available and FIG. 31B shows an example when only above neighboring samples are available and top-right is not available.

[0067] FIG. 32 shows examples of neighboring samples to derive LIC parameters.

DETAILED DESCRIPTION

[0068] Due to the increasing demand of higher resolution video, video coding methods and techniques are ubiquitous in modern technology. Video codecs typically include an electronic circuit or software that compresses or decompresses digital video, and are continually being improved to provide higher coding efficiency. A video codec converts uncompressed video to a compressed format or vice versa. There are complex relationships between the video quality, the amount of data used to represent the video (determined by the bit rate), the complexity of the encoding and decoding algorithms, sensitivity to data losses and errors, ease of editing, random access, and end-to-end delay (latency). The compressed format usually conforms to a standard video compression specification, e.g., the High Efficiency Video Coding (HEVC) standard (also known as H.265 or MPEG-H Part 2), the Versatile Video Coding (VVC) standard to be finalized, or other current and/or future video coding standards.

[0069] Embodiments of the disclosed technology may be applied to existing video coding standards (e.g., HEVC, H.265) and future standards to improve runtime performance. Section headings are used in the present document to improve readability of the description and do not in any way limit the discussion or the embodiments (and/or implementations) to the respective sections only.

1 Embodiments of cross-component prediction

[0070] Cross-component prediction is a form of the chroma-to-luma prediction approach that has a well-balanced trade-off between complexity and compression efficiency improvement.

1.1 Examples of the cross-component linear model (CCLM)

[0071] In some embodiments, and to reduce the cross-component redundancy, a cross-

component linear model (CCLM) prediction mode (also referred to as LM), is used in the JEM, for which the chroma samples are predicted based on the reconstructed luma samples of the same CU by using a linear model as follows:

[0072] $$pred_C(i,j) = \alpha \cdot rec_L{}'(i,j) + \beta \quad (1)$$

[0073] Here, $pred_C(i,j)$ represents the predicted chroma samples in a CU and $rec_L{}'(i,j)$ represents the downsampled reconstructed luma samples of the same CU for color formats 4:2:0 or 4:2:2 while $rec_L{}'(i,j)$ represents the reconstructed luma samples of the same CU for color format 4:4:4. CCLM parameters α and β are derived by minimizing the regression error between the neighboring reconstructed luma and chroma samples around the current block as follows:

[0074] $$\alpha = \frac{N \cdot \sum(L(n) \cdot C(n)) - \sum L(n) \cdot \sum C(n)}{N \cdot \sum(L(n) \cdot L(n)) - \sum L(n) \cdot \sum L(n)} \quad (2)$$

[0075] and $$\beta = \frac{\sum C(n) - \alpha \cdot \sum L(n)}{N}. \quad (3)$$

[0076] Here, $L(n)$ represents the down-sampled (for color formats 4:2:0 or 4:2:2) or original (for color format 4:4:4) top and left neighboring reconstructed luma samples, $C(n)$ represents the top and left neighboring reconstructed chroma samples, and value of N is equal to twice of the minimum of width and height of the current chroma coding block.

[0077] In some embodiments, and for a coding block with a square shape, the above two equations are applied directly. In other embodiments, and for a non-square coding block, the neighboring samples of the longer boundary are first subsampled to have the same number of samples as for the shorter boundary. FIG. 1 shows the location of the left and above reconstructed samples and the sample of the current block involved in the CCLM mode.

[0078] In some embodiments, this regression error minimization computation is performed as part of the decoding process, not just as an encoder search operation, so no syntax is used to convey the α and β values.

[0079] In some embodiments, the CCLM prediction mode also includes prediction between the two chroma components, e.g., the Cr (red-difference) component is predicted from the Cb (blue-difference) component. Instead of using the reconstructed sample signal, the CCLM Cb-to-Cr prediction is applied in residual domain. This is implemented by adding a weighted reconstructed Cb residual to the original Cr intra prediction to form the final Cr prediction:

[0080] $$pred_{Cr}^{*}(i,j) = pred_{Cr}(i,j) + \alpha \cdot resi_{Cb}{}'(i,j) \quad (4)$$

[0081] Here, $resi_{Cb}{}'(i,j)$ presents the reconstructed Cb residue sample at position (i,j).

[0082] In some embodiments, the scaling factor α may be derived in a similar way as in the CCLM luma-to-chroma prediction. The only difference is an addition of a regression cost relative to a default α value in the error function so that the derived scaling factor is biased towards a default value of −0.5 as follows:

[0083]
$$\alpha = \frac{N\cdot\sum(Cb(n)\cdot Cr(n))-\sum Cb(n)\cdot\sum Cr(n)+\lambda\cdot(-0.5)}{N\cdot\sum(Cb(n)\cdot Cb(n))-\sum Cb(n)\cdot\sum Cb(n)+\lambda} \quad (5)$$

[0084] Here, $Cb(n)$ represents the neighboring reconstructed Cb samples, $Cr(n)$ represents the neighboring reconstructed Cr samples, and λ is equal to $\sum(Cb(n)\cdot Cb(n)) \gg 9$.

[0085] In some embodiments, the CCLM luma-to-chroma prediction mode is added as one additional chroma intra prediction mode. At the encoder side, one more RD cost check for the chroma components is added for selecting the chroma intra prediction mode. When intra prediction modes other than the CCLM luma-to-chroma prediction mode is used for the chroma components of a CU, CCLM Cb-to-Cr prediction is used for Cr component prediction.

1.2 Examples of multiple model CCLM

[0086] In the JEM, there are two CCLM modes: the single model CCLM mode and the multiple model CCLM mode (MMLM). As indicated by the name, the single model CCLM mode employs one linear model for predicting the chroma samples from the luma samples for the whole CU, while in MMLM, there can be two models.

[0087] In MMLM, neighboring luma samples and neighboring chroma samples of the current block are classified into two groups, each group is used as a training set to derive a linear model (i.e., a particular α and β are derived for a particular group). Furthermore, the samples of the current luma block are also classified based on the same rule for the classification of neighboring luma samples.

[0088] FIG. 2 shows an example of classifying the neighboring samples into two groups. *Threshold* is calculated as the average value of the neighboring reconstructed luma samples. A neighboring sample with $Rec'_L[x,y] <= Threshold$ is classified into group 1; while a neighboring sample with $Rec'_L[x,y] > Threshold$ is classified into group 2.

[0089]
$$\begin{cases} Pred_c[x,y] = \alpha_1 \times Rec'_L[x,y] + \beta_1 & \text{if } Rec'_L[x,y] \le Threshold \\ Pred_c[x,y] = \alpha_2 \times Rec'_L[x,y] + \beta_2 & \text{if } Rec'_L[x,y] > Threshold \end{cases} \quad (6)$$

1.3 Examples of downsampling filters in CCLM

[0090] In some embodiments, and to perform cross-component prediction, for the 4:2:0

chroma format, where 4 luma samples corresponds to 1 chroma samples, the reconstructed luma block needs to be downsampled to match the size of the chroma signal. The default downsampling filter used in CCLM mode is as follows:

[0091]
$$Rec'_L[x,y] = \{2\times Rec_L[2x,2y] + 2\times Rec_L[2x,2y+1] + Rec_L[2x-1,2y] + Rec_L[2x+1,2y] + Rec_L[2x-1,2y+1] + Rec_L[2x+1,2y+1] + 4\} >> 3 \quad (7)$$

[0092] Here, the downsampling assumes the "type 0" phase relationship as shown in FIG. 3A for the positions of the chroma samples relative to the positions of the luma samples, e.g., collocated sampling horizontally and interstitial sampling vertically.

[0093] The exemplary 6-tap downsampling filter defined in (6) is used as the default filter for both the single model CCLM mode and the multiple model CCLM mode.

[0094] In some embodiments, and for the MMLM mode, the encoder can alternatively select one of four additional luma downsampling filters to be applied for prediction in a CU, and send a filter index to indicate which of these is used. The four selectable luma downsampling filters for the MMLM mode, as shown in FIG. 3B, are as follows:

[0095]
$$Rec'_L[x,y] = (Rec_L[2x,2y] + Rec_L[2x+1,2y] + 1) >> 1 \quad (8)$$

[0096]
$$Rec'_L[x,y] = (Rec_L[2x+1,2y] + Rec_L[2x+1,2y+1] + 1) >> 1 \quad (9)$$

[0097]
$$Rec'_L[x,y] = (Rec_L[2x,2y+1] + Rec_L[2x+1,2y+1] + 1) >> 1 \quad (10)$$

[0098]
$$Rec'_L[x,y] = (Rec_L[2x,2y] + Rec_L[2x,2y+1] + Rec_L[2x+1,2y] + Rec_L[2x+1,2y+1] + 2) >> 2 \quad (11)$$

1.4 Multi-directional LM (MDLM)

[0099] This existing implementation proposes multi-directional LM (MDLM). In MDLM, two additional CCLM modes are proposed: LM-A, where the linear model parameters are derived only based on the top-neighboring (or above-neighboring) samples as shown in FIG. 4A, and LM-L, where the linear model parameters are derived only based on the left-neighboring samples as shown in FIG. 4B.

1.5 Cross-component linear model simplification

[00100] This existing implementation proposes to replace the LMS algorithm of the linear model parameters α and β by a straight line equation, so called two-point method. The 2 points (couple of Luma and Chroma) (A, B) are the minimum and maximum values inside the set of

neighboring Luma samples as depicted in FIG. 5.

[00101] Herein, the linear model parameters α and β are obtained according to the following equation:

$$\alpha = \frac{y_B - y_A}{x_B - x_A} \text{ and } \beta = y_A - \alpha x_A.$$

[00102] In some embodiments, the division operation needed in the derivation of α is avoided and replaced by multiplications and shifts as below:

```
a = 0;
iShift    = 16;
int shift = (uiInternalBitDepth > 8) ? uiInternalBitDepth - 9 : 0;
int add   = shift ? 1 << (shift - 1) : 0;
int diff  = (MaxLuma- MinLuma + add) >> shift;
if (diff > 0)
{
  int div = ((MaxChroma- MinChroma) * g_aiLMDivTableLow[diff - 1] + 32768) >> 16;
  a    = (((MaxChroma- MinChroma) * g_aiLMDivTableHigh[diff - 1] + div + add) >> shift);
}
  b = MinLuma[1] - ((a * MinLuma[0]) >> iShift);
```

[00103] Herein, S is set equal to iShift, α is set equal to a and β is set equal to b. Furthermore, g_aiLMDivTableLow and g_aiLMDivTableHigh are two tables each with 512 entries, wherein each entry stores a 16-bit integer.

[00104] To derive the chroma predictor, as for the current VTM implementation, the multiplication is replaced by an integer operation as the following:

$$pred_C(i, j) = \left(\alpha \cdot rec_L'^{(i,j)}\right) \gg S + \beta$$

[00105] This implementation is also simpler than the current VTM implementation because shift S always has the same value.

1.6 Examples of CCLM in VVC

[00106] CCLM as in JEM is adopted in VTM-2.0, but MM-CCLM in JEM is not adopted in VTM-2.0. **MDLM** and simplified CCLM have been adopted into VTM-3.0.

1.7 Examples of Local Illumination Compensation in JEM

[00107] Local Illumination Compensation (LIC) is based on a linear model for illumination changes, using a scaling factor a and an offset b. And it is enabled or disabled adaptively for each inter-mode coded coding unit (CU).

[00108] When LIC applies for a CU, a least square error method is employed to derive the

parameters a and b by using the neighbouring samples of the current CU and their corresponding reference samples. More specifically, as illustrated in FIG. 32, the subsampled (2:1 subsampling) neighbouring samples of the CU and the corresponding pixels (identified by motion information of the current CU or sub-CU) in the reference picture are used. The IC parameters are derived and applied for each prediction direction separately.

[00109] When a CU is coded with 2Nx2N merge mode, the LIC flag is copied from neighbouring blocks, in a way similar to motion information copy in merge mode; otherwise, an LIC flag is signalled for the CU to indicate whether LIC applies or not.

[00110] When LIC is enabled for a pciture, addtional CU level RD check is needed to determine whether LIC is applied or not for a CU. When LIC is enabled for a CU, mean-removed sum of absolute diffefference (MR-SAD) and mean-removed sum of absolute Hadamard-transformed difference (MR-SATD) are used, instead of SAD and SATD, for integer pel motion search and fractional pel motion search, respectively.

[00111] To reduce the encoding complexity, the following encoding scheme is applied in JEM: LIC is disabled for the entire picture when there is no obvious illumination change between a current picture and its reference pictures. To identify this situation, histograms of a current picture and every reference picture of the current picture are calculated at the encoder. If the histogram difference between the current picture and every reference picture of the current picture is smaller than a given threshold, LIC is disabled for the current picture; otherwise, LIC is enabled for the current picture.

2 Examples of drawbacks in existing implementations

[00112] Current implementations introduce a two-point method to replace the LMS approach of LM mode in JEM. Although the new method decreases the number of additions and multiplications in CCLM, it introduces the following problems:

[00113] 1) Comparisons are introduced to find the minimum and maximum luma values, which are not friendly to a single instruction, multiple data (SIMD) software design.

[00114] 2) Two lookup-tables with 1024 entries in total storing 16-bit numbers are introduced, with a 2K ROM memory requirement that is not desirable in a hardware design.

3 Exemplary methods for cross-component prediction in video coding

[00115] Embodiments of the presently disclosed technology overcome drawbacks of existing implementations, thereby providing video coding with higher coding efficiencies and lower

computational complexity. Simplified linear model derivations for cross-component prediction, based on the disclosed technology, may enhance both existing and future video coding standards, is elucidated in the following examples described for various implementations. The examples of the disclosed technology provided below explain general concepts, and are not meant to be interpreted as limiting. In an example, unless explicitly indicated to the contrary, the various features described in these examples may be combined.

[00116] In the following examples and methods, the term "LM method" includes, but is not limited to, the LM mode in JEM or VTM, and MMLM mode in JEM, left-LM mode which only uses left neighboring samples to derive the linear model, the above-LM mode which only uses above neighboring samples to derive the linear model or other kinds of methods which utilize luma reconstruction samples to derive chroma prediction blocks. All LM modes which are not the LM-L nor the LM-A are called normal LM modes.

[00117] In the following examples and methods, Shift(x, s) is defined as $Shift(x,s) = (x + off) \gg s$, and SignShift(x, s) is defined as

$$SignShift(x,s) = \begin{cases} (x + off) \gg s & x \geq 0 \\ -((-x + off) \gg s) & x < 0 \end{cases}$$

[00118] Herein, *off* is an integer such as 0 or 2^{s-1}.

[00119] The height and width of a current chroma block are denoted *H* and *W*, respectively.

[00120] FIG. 6 shows an example of neighboring samples of the current chroma block. Let the coordinate of the top-left sample of the current chroma block be denoted as (x, y). Then, the neighboring chroma samples (as shown in FIG. 6) are denoted as:

[00121] A: Top sample at left: [x-1, y],

[00122] B: Top middle sample at left: [x-1, y+H/2-1],

[00123] C: Bottom middle sample at left: [x-1, y+H/2],

[00124] D: Bottom sample at left: [x-1, y+H-1],

[00125] E: Extended-bottom top sample at left: [x-1, y+H],

[00126] F: Extended-bottom top middle sample at left: [x-1, y+H+H/2-1],

[00127] G: Extended-bottom bottom middle sample at left: [x-1, y+H+H/2],

[00128] I: Extended-bottom bottom sample at left: [x-1, y+H+H-1],

[00129] J: Left sample at above: [x, y-1],

[00130] K: Left middle sample at above: [x+W/2-1, y-1],

[00131] L: Right middle sample at above: [x+W/2, y-1],

[00132] M: Right sample at above: [x+W-1, y-1],

[00133] N: Extended-above left sample at above: [x+W, y-1],

[00134] O: Extended-above left middle sample at above: [x+W +W/2-1, y-1],

[00135] P: Extended-above right middle sample at above: [x+W +W/2, y-1], and

[00136] Q: Extended-above right sample at above: [x+W +W-1, y-1].

[00137] Example 1. The parameters α and β in LM methods are derived from chroma samples at two or more specific positions.

a. The derivation is also dependent on the corresponding down-sampled luma samples of selected chroma samples. Alternatively, the derivation is also dependent on the corresponding luma samples of selected chroma samples such as when it is 4:4:4 color format.

b. For example, the parameters α and β in CCLM are derived from chroma samples at 2^S (e.g. S=2 or 3) positions, such as:

i. Position {A, D, J, M};

ii. Position {A, B, C, D, J, K, L, M};

iii. Position {A, I, J, Q};

iv. Position {A, B, D, I, J, K, M, Q};

v. Position {A, B, D, F, J, K, M, O};

vi. Position {A, B, F, I, J, K, O, Q};

vii. Position {A, C, E, I, J, L, N, Q};

viii. Position {A, C, G, I, J, L, P, Q};

ix. Position {A, C, E, G, J, L, N, P};

x. Position {A, B, C, D};

xi. Position {A, B, D, I};

xii. Position {A, B, D, F};

xiii. Position {A, C, E, I};

xiv. Position {A, C, G, I};

xv. Position {A, C, E, G};

xvi. Position {J, K, L, M};

xvii. Position {J, K, M, Q};

xviii. Position {J, K, M, O};
xix. Position {J, K, O, Q};
xx. Position {J, L, N, Q};
xxi. Position {J, L, P, Q};
xxii. Position {J, L, N, P};
xxiii. Position {A, B, C, E, E, F, G, I};
xxiv. Position {J, K, L, M, N, O, P, Q};

c. For example, the parameters α and β in CCLM are derived from chroma samples at:
    i. Any combination between {A, B, C, D, E, F, G, I} and {J, K, L, M, N, O, P, Q} such as
        (a) Position A and J;
        (b) Position B and K;
        (c) Position C and L;
        (d) Position D and M;
        (e) Position E and N;
        (f) Position F and O;
        (g) Position G and P;
        (h) Position I and Q;
    ii. Any two different positions fetched from {A, B, C, D, E, F, G, }
        (a) Position A and B;
        (b) Position A and C;
        (c) Position A and D;
        (d) Position A and E;
        (e) Position A and F;
        (f) Position A and G;
        (g) Position A and I;
        (h) Position D and B;
        (i) Position D and C;
        (j) Position E and B;
        (k) Position E and C;

(l) Position I and B;
(m) Position I and C;
(n) Position I and D;
(o) Position I and E;
(p) Position I and F;
(q) Position I and G;

iii. Any two different positions fetched from {J, K, L, M, N, O, P, Q}
(a) Position J and K;
(b) Position J and L;
(c) Position J and M;
(d) Position J and N;
(e) Position J and O;
(f) Position J and P;
(g) Position J and Q;
(h) Position M and K;
(i) Position M and L;
(j) Position N and K;
(k) Position N and L;
(l) Position Q and K;
(m) Position Q and L;
(n) Position Q and M;
(o) Position Q and N;
(p) Position Q and O;
(q) Position Q and P;
(r) Position Q and Q;

iv. In one example, if the two selected positions have identical luma value, more positions may be further checked.

d. For example, not all available chroma samples are searched to find the minimum and maximum luma values to derive the parameters α and β in CCLM with the two-point method.

i. One chroma sample out of K chroma samples (and their corresponding down-sampled luma samples) are included in the searching set. K may be 2, 4, 6 or 8.
   (a) For example, if Rec[x,y] is an above neighboring sample, it is included in the searching set only if x%K==0. If Rec[x,y] is a left neighboring sample, it is included in the searching set only if y%K==0.
ii. Only Chroma samples at specific positions such as defined in 1.a.i~1.a.xxiv are included in the searching set.

e. For mode LM-L, all selected samples must be left-neighboring samples.
f. For mode LM-A, all selected samples must be above-neighboring samples.
g. The selected positions can be fixed, or they can be adaptive.
   i. In one example, which positions are selected may depend on the width and height of the current chroma block;
   ii. In one example, which positions are selected may be signaled from the encoder to the decoder, such as in VPS/SPS/PPS/slice header/tile group header/tile/CTU/CU/PU.
h. The selected chroma samples are used to derive the parameters α and β with the least mean square method as shown in Eq(2) and Eq(3). In Eq(2) and Eq(3), N is set to be the number of the selected samples.
i. A pair of selected chroma samples are used to derive the parameters α and β with the two-point method.
j. In one example, how to select samples may depend on the availability of the neighboring blocks.
   i. For example, positions A, D, J and M are selected if both the left and the above neighboring blocks are available; position A and D are selected if only the left neighboring block is available; and position J and M are selected if only the above neighboring block is available.

[00138] Example 2. Sets of parameters in CCLM mode can be firstly derived and then combined to form the final linear model parameter used for coding one block. Suppose α_1 and β_1 are derived from a group of chroma samples at specific positions denoted as Group 1, α_2 and β_2

are derived from a group of chroma samples at specific positions denoted as Group 2, … , α_N and β_N are derived from a group of chroma samples at specific positions denoted as Group N, then the final α and β can be derived from (α_1, β_1), … (α_N, β_N).

a. In one example, α is calculated as the average of α_1, … α_N and β is calculated as the average of β_1, … β_N.
   i. In one example, α=SignShift(α_1+ α_2, 1), β=SignShift(β_1+ β_2, 1).
   ii. In one example, α=Shift(α_1+ α_2, 1), β=Shift(β_1+ β_2, 1).
   iii. If (α_1, β_1) and (α_2, β_2) are with different precision, for example, To get a chroma prediction CP from its corresponding down-sampled luma sample LR, it is calculated as

   $CP = SignShift(\alpha_1 \times LR + \beta_1, Sh_1)$ with (α_1, β_1), but

   $CP = SignShift(\alpha_2 \times LR + \beta_2, Sh_2)$ with (α_2, β_2) Sh_1 is not equal to Sh_2, then the parameters need to be shifted before being combined. Suppose $Sh_1 > Sh_2$, then before combining, the parameters should be shifted as:
      (a) $\alpha_1 = SignShift(\alpha_1, Sh_1 - Sh_2), \beta_1 = SignShift(\beta_1, Sh_1 - Sh_2)$. Then the final precision is as (α_2, β_2).
      (b) $\alpha_1 = Shift(\alpha_1, Sh_1 - Sh_2), \beta_1 = Shift(\beta_1, Sh_1 - Sh_2)$. Then the final precision is as (α_2, β_2).
      (c) $\alpha_2 = \alpha_2 << (Sh_1 - Sh_2), \beta_2 = \beta_2 << (Sh_1 - Sh_2)$. Then the final precision is as (α_1, β_1).

b. Some examples of positions in Group 1 and Group 2:
   i. Group 1: Position A and D, Group 2: Position J and M.
   ii. Group 1: Position A and I, Group 2: Position J and Q.
   iii. Group 1: Position A and D, Group 2: Position E and I, where there are two groups are used for mode LM-L.
   iv. Group 1: Position J and M, Group 2: Position N and Q, where there are two groups are used for mode LM-A.
   v. Group 1: Position A and B, Group 2: Position C and D, where there are two groups are used for mode LM-L.

vi. Group 1: Position J and K, Group 2: Position L and M, where there are two groups are used for mode LM-A.

[00139] Example 3. Suppose two chroma sample values denoted as C0 and C1, and their corresponding luma sample values denoted as L0 and L1 (L0 < L1) are inputs. The two-point method can derive α and β with the input as

$$\alpha = \frac{C1 - C0}{L1 - L0} \text{ and } \beta = C0 - \alpha L0.$$

[00140] The bit depths of luma samples and chroma samples are denoted BL and BC. One or more simplifications for this implementation include:

a. α is output as 0 if L1 is equal to L0. Alternatively, when L1 is equal to L0, a certain intra prediction mode (e.g., DM mode, DC or planar) is used instead of using CCLM mode to derive the prediction block.

b. The division operation is replaced by other operations with no lookup table. log2 operation may be implemented by checking position of the most significant digit.

i. $\alpha = Shift(C1 - C0, Floor(\log_2(L1 - L0))$ or

$\alpha = SignShift(C1 - C0, Floor(\log_2(L1 - L0))$

ii. $\alpha = Shift(C1 - C0, Ceiling(\log_2(L1 - L0))$ or

$\alpha = SignShift(C1 - C0, Ceiling(\log_2(L1 - L0))$

iii. Example i or Example ii may be selected based on the value of L1-L0.

(a) For example, Example i is used if L1-L0<T, otherwise Example ii is used. For example, T can be

$(Floor(\log_2(L1 - L0)) + Ceiling(\log_2(L1 - L0)))/2$

(b) For example, Example i is used if $3 \times (L1 - L0) < 2^{Floor(Log_2(L1-L0))+2}$, otherwise Example ii is used.

(c) For example, Example i is used if $(L1 - L0)^2 < 2^{2 \times Floor(Log_2(L1-L0))+1}$, otherwise Example ii is used.

c. The division operation is replaced by one lookup table denoted as M[k].

i. The size of the lookup table denoted as V is less than 2^P, where P is an integer number such as 5, 6, or 7.

ii. Each entry of the lookup table stores an F-bit integer number, e.g., F= 8 or 16.
   (a) In one example, M[k-Z]=((1<<S)+Off)/k, where S is an integer defining the precision, e.g., S=F. Off is an offset, e.g., Off=(k+Z)>>1. Z defines the beginning value of the table, e.g., Z=1, or Z=8, or Z= 32. A valid key k inquiring the table must satisfy k >= Z.
iii. k=Shift(L1-L0, W) is used as the key to inquire the lookup table.
   (a) In one example, W depends on BL, V and Z.
   (b) In one example, W also depends on the value of L1-L0.
iv. If k is not a valid key to inquire the lookup table (k-Z < 0 or k-Z >=V), α is output as 0.
v. For example,
   $\alpha = Shift((C1-C0)\times M[k-Z], D)$, or
   $\alpha = SignShift((C1-C0)\times M[k-Z], D)$
vi. To get a chroma prediction CP from its corresponding (e.g., down-sampled for 4:2:0) luma sample LR, it is calculated as
   $CP = SignShift(\alpha \times LR + \beta, Sh)$, or
   $CP = Shift(\alpha \times LR + \beta, Sh)$
vii. Sh can be a fixed number, or it may depend on the values of C0, C1, L0, L1 used to calculated α and β.
   (a) Sh may depend on BL, BC, V, S and D.
   (b) D may depend on Sh.
viii. The size of the lookup table denoted as V is equal to 2^P, where P is an integer number such as 5, 6, 7 or 8. Alternatively, V is set to 2^P - M (e.g., M is equal to 0).
ix. Suppose $\alpha = P/Q$ (e.g. Q = L1-L0, P = C1-C0, or they are derived in other ways), then α is calculated with the lookup table as
   $\alpha = Shift(P\times M[k-Z], D)$ or $\alpha = SignShift(P\times M[k-Z], D)$, where k is the key (index) to inquire an entry in the lookup table.

(a) In one example, k is derived from Q with a function: k= f(Q).

(b) In one example, k is derived from Q and P, with a function: k= f(Q, P).

(c) In one example, k is valid in a specific range [kMin, kMax]. For example, kMin = Z and kMax = V+Z.

(d) In one example, k=Shift(Q, W),

a. W may depend on BL, V and Z.

b. W may depend on the value of Q.

c. In one example, when k is calculated as Shift(Q, W), then α is calculated with the lookup table as

$\alpha = (Shift(P \times M[k-Z], D)) << W$ or

$\alpha = (SignShift(P \times M[k-Z], D)) << W$

(e) In one example, k is derived in different ways with different values of Q.

a. For example, k= Q when Q<= kMax, and k = Shift(Q, W) when Q > kMax. For example, W is chosen as the smallest positive integer that makes Shift(Q, W) no greater than kMax.

b. For example, k = Min(kMax, Q).

c. For example, k = Max(kMin, Min(kMax, Q)).

(f) In one example, when Q < 0, -Q is used to replace Q in the calculation. Then $-\alpha$ is output.

(g) In one example, when Q is equal to 0, then α is set to be a default value such as 0 or 1.

(h) In one example, when Q is equal to 2^E E>=0, then $\alpha = Shift(P, E)$ or $\alpha = SignShift(P, E)$.

d. All operations to derive the LM parameters must be within K bits, K can be 8, 10, 12, 16, 24 or 32.

i. If an intermedia variable may exceed the range represented by the constrained bits, it should be clipped or right shifted to be within the constrained bits.

[00141] Example 4. One single chroma block may use multiple linear models and the selection of multiple linear model is dependent on the position of chroma samples within the chroma block.

a. In one example, LM-L and LM-A mode can be combined in a single chroma block.
b. In one example, some samples are predicted by LM-L mode and other samples are predicted by LM-A mode.
    i. Fig. 7 shows an example. Suppose the top-left sample is at position (0,0). Samples at position (x,y) with x>y (or x>=y) are predicted by LM-A, and other samples are predicted by LM-L.
c. Suppose the prediction with LM-L and LM-A for a sample at position (x,y) are denoted as P1(x,y) and P2(x,y), respectively, then the final prediction P(x,y) is calculated as a weighted sum of P1(x,y) and P2(x,y).
    i. P(x,y)=w1*P1(x,y)+w2*P2(x,y)
        (a) w1+w2=1.
    ii. P(x,y)=(w1*P1(x,y)+w2*P2(x,y)+Offset)>>shift, where offset may be 0 or 1<<(shift-1), and shift is an integer such as 1, 2, 3….
        (a) w1+w2=1<<shift.
    iii. P(x,y)=(w1*P1(x,y)+((1<<shift)-w1)*P2(x,y)+Offset)>>shift, where offset may be 0 or 1<<(shift-1), and shift is an integer such as 1, 2, 3….
    iv. w1 and w2 may depend on the position (x,y)
        (a) For example, w1 > w2 (e.g. w1=3, w2=1) if x<y,
        (b) For example, w1 < w2 (e.g. w1=1, w2=3) if x>y,
        (c) For example, w1 = w2 (e.g. w1=2, w2=2) if x==y,
        (d) For example, w1 – w2 increases if y-x increases when x <y,
        (e) For example, w2 – w1 increases if x-y increases when x >y.

[00142] Example 5. It is proposed that the neighboring samples (including chroma samples and their corresponding luma samples, which may be down-sampled) are divided into N groups. The maximum luma value and minimum luma value for the k-th group (with k = 0, 1,…., N-1) is denoted as $MaxL_k$ and $MinL_k$, and their corresponding chroma values are denoted as $MaxC_k$ and $MinC_k$, respectively.

[00143] a. In one example, MaxL is calculated as MaxL=f1($MaxL_{S0}$, $MaxL_{S1}$, …, $MaxL_{Sm}$); MaxC is calculated as MaxC=f2($MaxC_{S0}$, $MaxC_{S1}$,…$MaxC_{Sm}$); MinL is calculated as MinL=f3($MinL_{S0}$, $MinL_{S1}$,…$MinL_{Sm}$). MinC is calculated as MinC=f3($MinC_{S0}$, $MinC_{S1}$, …, $MinC_{Sm}$). f1, f2, f3 and f4 are functions. The two-point method derives α and β with the input as:

$$\alpha = \frac{MaxC - MinC}{MaxL - MinL}$$
$$\beta = MinC - \alpha MinL$$

[00144] i. In one example, f1, f2, f3, f4 all represent the averaging function.

[00145] ii. S0, S1,…Sm are indices of selected groups which are used to calculate α and β.

[00146] (1) For example, all groups are used, e.g., S0=0, S1=1,… Sm=N-1.

[00147] (2) For example, two groups are used, e.g., m=1, S0=0, S1 = N-1.

[00148] (3) For example, not all groups are used, e.g. m < N-1, S0=0, S1=2, S3= 4, …

[00149] b. In one example, samples (or down-sampled samples) located at above rows may be classified to one group and samples (or down-sampled samples) located at left columns of a block may be classified to another group.

[00150] c. In one example, samples (or down-sampled samples) are classified based on their locations or coordinates.

[00151] i. For examples, samples may be classified into two groups.

[00152] (1) For a sample with coordinate (x,y) located at above rows, it is classified into group S0 if x%P = Q, where P and Q are integers, e.g. P = 2, Q =1, P=2, Q = 0 or P = 4, Q = 0; Otherwise, it is classified into group S1.

[00153] (2) For a sample with coordinate (x,y) located at left columns, it is classified into group S0 if y%P = Q, where P and Q are integers, e.g. P = 2, Q =1, P=2, Q = 0 or P = 4, Q = 0; Otherwise, it is classified into group S1.

[00154] (3) Only samples in one group, such as S0, are used to find MaxC and MaxL. For example, MaxL= MaxLS0 and MaxC= MaxCS0.

[00155] d. In one example, only partial of neighboring samples (or down-sampled samples) are used for divided to N groups.

[00156] e. The number of groups (e.g., N) and/or the selected group indices and/or functions (f1/f2/f3/f4) may be pre-defined or signaled in SPS/VPS/PPS/picture header/slice header/tile group header/LCUs/LCU/CUs.

[00157] f. In one example, how to select the samples for each group may depend the availability of neighboring blocks.

[00158] i. For example, $MaxL_0$ /$MaxC_0$ and $MinL_0$ /$MinC_0$ are found from position A and D; $MaxL_1$ /$MaxC_1$ and $MinL_1$ /$MinC_1$ are found from position J and M, then MaxL= ($MaxL_0$ + $MaxL_1$)/ 2, MaxC= ($MaxC_0$ + $MaxC_1$)/ 2, MinL= ($MinL_0$ + $MinL_1$)/ 2, MinC= ($MinC_0$ + $MinC_1$)/ 2, when both the left and the above neighboring blocks are available.

[00159] ii. For example, MaxL /MaxC and MinL /MinC are directly found from position A and D when only the left neighboring block is available.

[00160] (1) Alternatively, α and β are set equal to some default values if the above neighboring block is not available. For example, α = 0 and β=1<<(bitDepth -1), where bitDepth is the bit depth of the chroma samples.

[00161] iii. For example, MaxL /MaxC and MinL /MinC are directly found from position J and M when only the above neighboring block is available.

[00162] (1) Alternatively, α and β are set equal to some default values if the left neighboring block is not available. For example, α = 0 and β=1<<(bitDepth -1), where bitDepth is the bit depth of the chroma samples.

[00163] g. In one example, how to select the samples for each group may depend the width and height of the block.

[00164] h. In one example, how to select the samples for each group may depend on the values of samples.

[00165] i, In one example, the two samples with the largest luma value and minimum luma value are picked out to be in a first group. And all other samples are in a second group.

[00166] Example 6. It is proposed that whether and how to apply LM-L and LM-A mode may depend on the width (W) and height (H) of the current block.

[00167] (a) For example, LM-L cannot be applied if W > K×H. e.g., K = 2.

[00168] (b) For example, LM-A cannot be applied if H > K×W. e.g., K = 2.

[00169] (c) If one of LM-L and LM-A cannot be applied, the flag to indicate whether LM-

L or LM-A is used should not be signaled.

[00170] Example 7. A flag is signaled to indicate whether CCLM mode is applied. The context used in arithmetic coding to code the flag may depend on whether the top-left neighboring block as shown in FIG. 8 applies CCLM mode or not.

[00171] (a) In one example, a first context is used if the top-left neighboring block applies CCLM mode; and a second context is used if the top-left neighboring block does not apply CCLM mode.

[00172] (b) In one example, if the top-left neighboring block is not available, it is considered as not applying CCLM mode.

[00173] (c) In one example, if the top-left neighboring block is not available, it is considered as applying CCLM mode.

[00174] (d) In one example, if the top-left neighboring block is not intra-coded, it is considered as not applying CCLM mode.

[00175] (e) In one example, if the top-left neighboring block is not intra-coded, it is considered as applying CCLM mode.

[00176] Example 8. Indications or codewords of DM and LM modes may be coded in different orders from sequence to sequence/picture to picture/tile to tile/block to block.

[00177] (a) The coding order of indications of LM and DM (e.g., firstly code whether it is LM mode, if not, then code whether it is DM mode; or firstly code whether it is DM mode, if not, then code whether it is LM mode) may be depend on the mode information of one or multiple neighboring blocks.

[00178] (b) In one example, when the top-left block of the current block is available and coded with LM mode, then the indication of LM mode is firstly coded.

[00179] (c) Alternatively, when the top-left block of the current block is available and coded with DM mode, then the indication of DM mode is firstly coded.

[00180] (d) Alternatively, when the top-left block of the current block is available and coded with non-LM (e.g., DM mode or other intra prediction modes excluding LM), then the indication of DM mode is firstly coded.

[00181] (e) In one example, indications of the order may be signaled in in SPS/VPS/PPS/picture header/slice header/tile group header/LCUs/LCU/CUs.

[00182] Example 9. In above examples, samples (or down-sampled samples) may be located

beyond the range of 2×W above neighboring samples or 2×H left neighboring samples as shown in FIG. 6.

[00183] (a) With LM mode or LM-L mode, it may use a neighboring sample RecC[x-1, y+d], where d is in the range of [T, S]. T may be smaller than 0, and S may be larger than 2H-1. For example, T = -4 and S = 3H. In another example, T = 0, S = max(2H, W+H). In still another example, T = 0 and S = 4H.

[00184] (b) With LM mode or LM-A mode, it may use a neighboring sample RecC[x+d, y], where d is in the range of [T, S]. T may be smaller than 0, and S may be larger than 2W-1. For example, T = -4 and S = 3W. In another example, T = 0, S = max(2W, W+H). In still another example, T = 0 and S = 4W.

[00185] Example 10. In one example, the chroma neighboring samples and their corresponding luma samples (may be down-sampled) are down-sampled before deriving the linear model parameters α and β as disclosed in Examples 1-7. Suppose the width and height of the current chroma block is W and H.

[00186] (a) In one example, whether and how to conduct down-sampling may depend on W and H.

[00187] (b) In one example, the number of neighboring samples used to derive the parameters left to the current block, and the number of neighboring samples used to derive the parameters above to the current block should be the same after the down-sampling process.

[00188] (c) In one example, the chroma neighboring samples and their corresponding luma samples (may be down-sampled) are not down-sampled if W is equal to H.

[00189] (d) In one example, the chroma neighboring samples and their corresponding luma samples (may be down-sampled) left to the current block are down-sampled if W < H.

[00190] (i) In one example, one chroma sample in each H/W chroma samples are picked to be used for deriving α and β. Other chroma samples are discarded. For example, suppose R[0, 0] represents the top-left sample of the current block, then R[-1, K*H/W], K from 0 to W-1, are picked to be used for deriving α and β.

[00191] (e) In one example, the chroma neighboring samples and their corresponding luma samples (may be down-sampled) above to the current block are down-sampled if W > H.

[00192] (ii) In one example, one chroma sample in each W/H chroma samples are picked to be used for deriving α and β. Other chroma samples are discarded. For example,

suppose R[0, 0] represents the top-left sample of the current block, then R[K*W/H, -1], K from 0 to H-1, are picked to be used for deriving α and β.

[00193] (ii) FIG. 9 shows examples of samples to be picked up when position D and position M in FIG. 6 are used to derive α and β, and down-sampling performed when W>H.

[00194] Example 11. Neighboring downsampled/originally reconstructed samples and/or downsampled/originally reconstructed samples may be further refined before being used in the linear model prediction process or cross-color component prediction process.

[00195] (a) "To be refined" may refer to a filtering processing.

[00196] (b) "To be refined" may refer to any non-linear processing

[00197] (c) It is proposed that several neighbouring samples (including chroma samples and their corresponding luma samples, which may be down-sampled) are picked up to calculate C1, C0, L1 and L0, in order to derive α and β, such as $\alpha = (C1-C0) / (L1-L0)$ and $\beta = C0 - \alpha L0$.

[00198] (d) In one example, S neighboring luma samples (maybe down-sampled) denoted as Lx1, Lx2,…, LxS, and their corresponding chroma samples denoted as Cx1, Cx2, … CxS are used to derive C0 and L0, and T neighboring luma samples (maybe down-sampled) denoted as Ly1, Ly2,…, LyT, and their corresponding chroma samples denoted as Cy1, Cy2, … CyT are used to derive C1 and L1 as:

[00199] (i) C0=f0(Cx1, Cx2, … CxS), L0=f1(Lx1, Lx2, … LxS), C1=f2(Cy1, Cy2, … CyT), L1=f4(Ly1, Ly2, … LyT). f0, f1, f2 and f3 are any functions.

[00200] (ii) In one example, f0 is identical to f1.

[00201] (iii) In one example, f2 is identical to f3.

[00202] (iv) In one example, f0 f1 f2 f3 are identical.

[00203] 1. For example, they are all the averaging function.

[00204] (v) In one example, S is equal to T.

[00205] 1. In one example, the set {x1,x2,…xS} is identical to the set {y1, y2, …, yT}.

[00206] (vi) In one example, Lx1, Lx2,…, LxS are chosen as the smallest S luma samples of a group of luma samples.

[00207] 1. For example, the group of luma samples includes all neighboring samples used in VTM-3.0 to derive CCLM linear parameters.

[00208] 2. For example, the group of luma samples includes partial neighboring samples used in VTM-3.0 to derive CCLM linear parameters.

[00209] a. For example, the group of luma samples includes four samples as shown in Fig. 2-5.

[00210] (vii) In one example, Ly1, Ly2,…, LyS are chosen as the largest S luma samples of a group of luma samples.

[00211] 1. For example, the group of luma samples includes all neighboring samples used in VTM-3.0 to derive CCLM linear parameters.

[00212] 2. For example, the group of luma samples includes partial neighboring samples used in VTM-3.0 to derive CCLM linear parameters.

[00213] a. For example, the group of luma samples includes four samples as shown in FIGS. 2-5.

[00214] Example 12. It is proposed to select other neighboring or downsampled neighboring samples based on the largest neighboring or downsampled neighboring sample in a given set of neighboring or downsampled neighboring samples.

[00215] (a) In one example, denote the largest neighboring or downsampled neighboring sample is located at position (x0, y0). Then samples in the region (x0-d1, y0), (x0, y0-d2), (x0+d3, y0), (x0, y0+d4) may be utilized to select other samples. Integers {d1, d2, d3, d4} may depend on the position (x0, y0). For example, if (x0, y0) is left to the current block, then d1 = d3 = 1 and d2=d4=0. If (x0,y0) is above to the current block, then d1 = d3=0 and d2=d4 = 1.

[00216] (b) In one example, denote the smallest neighboring or downsampled neighboring sample is located at position (x1, y1). Then samples in the region (x1-d1, y1), (x1, y1-d2), (x1+d3, y1), (x1, y1+d4) may be utilized to select other samples. Integers {d1, d2, d3, d4} may depend on the position (x1, y1). For example, if (x1, y1) is left to the current block, then d1 = d3 = 1 and d2=d4=0. If (x1,y1) is above to the current block, then d1 = d3=0 and d2=d4 = 1.

[00217] (c) In one example, the above samples are representing samples of one color component (e.g., luma color component). Samples used in CCLM/cross-color component process may be derived by corresponding coordinates of a second color component.

[00218] (d) Similar way can be used to derive smallest samples.

[00219] Example 13: In above examples, luma and chroma may be switched. Alternatively, luma color component may be replaced by the main color component (e.g., G), and chroma color

component may be replaced by dependent color component (e.g., B or R).

[00220] Example 14. Selection of locations of chroma samples (and/or corresponding luma samples) may depend on the coded mode information.

[00221] (a) Alternatively, furthermore, it may depend on the availability of neighboring samples, such as whether left column or above row or above-right row or below-left column is available. FIG. 10 depicts the concepts of left column/above row/above-right row/below-left column relative to a block.

[00222] (b) Alternatively, furthermore, it may depend on the availability of samples located at certain positions, such as whether the 1st top-right sample and/or 1st below-left sample is available.

[00223] (c) Alternatively, furthermore, it may depend on block dimensions.

[00224] (i) Alternatively, furthermore, it may depend on the ratio between width and height of current chroma (and/or luma) block.

[00225] (ii) Alternatively, furthermore, it may depend on whether the width and/or height is equal to K (e.g., K = 2).

[00226] (d) In one example, when the current mode is a normal LM mode, the following ways may be applied to select chroma samples (and/or luma samples downsampled or non-downsampled):

[00227] (i) If both left column and above row are available, two samples of left column and two of above row may be selected. They may be located at (suppose the top-left coordinate of the current block is (x, y)):

[00228] 1. (x-1, y), (x, y-1), (x-1, y+H-1) and (x + W-1, y-1)

[00229] 2. (x-1, y), (x, y-1), (x-1, y + H - H/W -1) and (x + W-1, y-1). For example, when H is larger than W.

[00230] 3. (x-1, y), (x, y-1), (x-1, y + H -1) and (x + W - W/H-1, y-1). For example, when H is smaller than W.

[00231] 4. (x-1, y), (x, y-1), (x-1, y + H – max(1, H/W)) and (x + W-max(1, W/H), y-1).

[00232] (ii) If only above row is available, samples are only selected from the above row.

[00233] 1. For example, four samples of above row may be selected.

[00234] 2. For example, two samples may be selected.

[00235] 3. How to select the samples may depend on the width/height. For example, four samples are selected when W>2 and two samples are selected when W=2.

[00236] 4. The selected samples may be located at (suppose the top-left coordinate of the current block is (x, y)):

[00237] a. (x, y-1), (x + W/4, y-1), (x + 2*W/4, y-1), (x + 3*W/4, y – 1)

[00238] b. (x, y-1), (x + W/4, y-1), (x + 3*W/4, y – 1), (x + W-1, y -1)

[00239] c. (x, y-1), (x + (2W)/4, y-1), (x + 2*(2W)/4, y-1), (x + 3*(2W)/4, y – 1).

[00240] For example, when top-right row is available, or when 1st top-right sample is available.

[00241] d. (x, y-1), (x + (2W)/4, y-1), (x + 3*(2W)/4, y – 1), (x + (2W)-1, y -1).

[00242] For example, when top-right row is available, or when 1st top-right sample is available.

[00243] (iii) If only left column is available, samples are only selected from the left column.

[00244] 1. For example, four samples of left column may be selected;

[00245] 2. For example, two samples of left column may be selected;

[00246] 3. How to select the samples may depend on the width/height. For example, four samples are selected when H>2 and two samples are selected when H=2.

[00247] 4. The selected samples may be located at:

[00248] a. (x-1, y), (x -1, y + H/4), (x -1, y + 2*H/4), (x -1, y + 3*H/4)

[00249] b. (x-1, y), (x - 1, y+ 2*H/4), (x -1, y + 3*H/4) , (x -1, y + H-1)

[00250] c. (x-1, y), (x -1, y + (2H)/4), (x -1, y + 2*(2H)/4), (x -1, y + 3*(2H)/4).

[00251] For example, when below-left column is available, or when 1st below-left sample is available.

[00252] d. (x-1, y), (x - 1, y+ 2*(2H)/4), (x -1, y + 3*(2H)/4) , (x -1, y + (2H)-1).

[00253] For example, when below-left column is available, or when 1st below-left sample is available.

[00254] (iv) For above examples, only two of the four samples may be selected.

[00255] (e) In one example, when the current mode is the LM-A mode, it may choose samples according to Example 11(d)(ii).

[00256] (f) In one example, when the current mode is the LM-L mode, it may choose samples according to Example 11(d)(iii).

[00257] (g) The luma selected samples (e.g., according to the selected chroma locations) may be grouped to 2 groups, one is with the largest value and smallest value of all selected samples, the other group is with all remaining samples.

[00258] (i) The two maximum values of 2 groups are averaged as the maximum value in the 2-point method; the two minimum values of 2 groups are averaged as the minimum value in the 2-point method to derive LM parameters.

[00259] (ii) When there are only 4 samples selected, two larger sample values are averaged, two smaller sample values are averaged, and averaged values are used as the input to the 2-point method to derive LM parameters.

[00260] Example 15. In above examples, luma and chroma may be switched. Alternatively, luma color component may be replaced by the main color component (e.g., G), and chroma color component may be replaced by dependent color component (e.g., B or R).

[00261] Example 16. It is proposed to select the above neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) based on a first position offset value (denoted as F) and a step value (denoted as S). Suppose the width of available above neighbouring samples to be used is W.

a. In one example, W may be set to the width of current block.
b. In one example, W may be set to (L* width of current block) wherein L is an integer value.
c. In one example, when both above and left blocks are available, W may be set to the width of current block.
    i. Alternatively, when the left block is NOT available, W may be set to (L* width of current block) wherein L is an integer value.
    ii. In one example, L may be dependent on the availability of top-right block. Alternatively, L may depend on the availability of one top-left sample.
d. In one example, W may depend on the coded mode.
    i. In one example, W may be set to the width of current block if the current block is coded as LM mode;

ii. W may be set to (L* width of current block) wherein L is an integer value if the current block is coded as LM-A mode.

(a) L may be dependent on the availability of top-right block. Alternatively, L may depend on the availability of one top-left sample.

e. Suppose the top-left coordinate of the current block is (x0, y0), then the above neighbouring samples at positions (x0+F+K×S, y0-1) with K =0, 1, 2,…kMax are selected.

f. In one example, F= W/P. P is an integer.

i. For example, $P = 2^i$, where i is an integer such as 1 or 2.

ii. Alternatively, F=W/P+*offset*.

g. In one example, S= W/Q. Q is an integer.

i. For example, $Q = 2^j$, where j is an integer such as 1 or 2.

h. In one example, F= S/R. R is an integer.

i. For example, $R = 2^m$, where m is an integer such as 1 or 2.

i. In one example, S= F/Z. Z is an integer.

i. For example, $Z = 2^n$, where n is an integer such as 1 or 2.

j. kMax and/or F and/or S and/or *offset* may depend on the prediction mode (such as LM, LM-A or LM-L) of the current block;

k. kMax and/or F and/or S and/or *offset* may depend on the width and/or height of the current block.

l. kMax and/or F and/or S and/or *offset* may depend on availability of neighbouring samples.

m. kMax and/or F and/or S and/or *offset* may depend on W.

n. For example, kMax=1, F= W/4, S= W/2, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM coded, both the left and above neighbouring samples are available, and W>=4.

o. For example, kMax=3, F= W/8, S= W/4, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM coded, only above neighbouring samples are available, and W>=4.

p. For example, kMax=3, F= W/8, S= W/4, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM-A coded and W>=4.
q. For example, kMax=1, F= 0, S= 1, *offset* = 0. Alternatively, furthermore, the settings are done if W is equal to 2.

[00262] Example 17. It is proposed to select the left neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) based on a first position offset value (denoted as F) and a step value (denoted as S). Suppose the height of available left neighbouring samples to be used is H.

a. In one example, H may be set to the height of current block.
b. In one example, H may be set to (L* height of current block) wherein L is an integer value.
c. In one example, when both above and left blocks are available, H may be set to the height of current block.
    i. Alternatively, when the above block is NOT available, H may be set to (L* height of current block) wherein L is an integer value.
    ii. In one example, L may be dependent on the availability of below-left block. Alternatively, L may be dependent on the availability of one below-left sample.
    iii. Alternatively, H may be set to (height of current block + width of the current block) if the required above-right neighbouring blocks are available.
        (a) In one example, same H above neighbouring samples are picked for LM-A mode and LM mode when left neighbouring samples are unavailable.
d. In one example, H may depend on the coded mode.
    i. In one example, H may be set to the height of current block if the current block is coded as LM mode;
    ii. W may be set to (L* height of current block) wherein L is an integer value if the current block is coded as LM-L mode.

(a) L may be dependent on the availability of below-left block. Alternatively, L may depend on the availability of one top-left sample.

(b) Alternatively, W may be set to (height of current block + width of the current block) if the required below-left neighbouring blocks are available.

(c) In one example, same W left neighbouring samples are picked for LM-L mode and LM mode when above neighbouring samples are unavailable.

e. Suppose the top-left coordinate of the current block is (x0, y0), then the left neighbouring samples at positions (x0-1, y0+F+K×S) with K =0, 1, 2,… kMax are selected.

f. In one example, F= H/P. P is an integer.

   i. For example, $P = 2^i$, where i is an integer such as 1 or 2.

   ii. Alternatively, F=H/P+*offset*.

g. In one example, S= H/Q. Q is an integer.

   i. For example, $Q = 2^j$, where j is an integer such as 1 or 2.

h. In one example, F= S/R. R is an integer.

   i. For example, $R = 2^m$, where m is an integer such as 1 or 2.

i. In one example, S= F/Z. Z is an integer.

   i. For example, $Z = 2^n$, where n is an integer such as 1 or 2.

j. kMax and/or F and/or S and/or *offset* may depend on the prediction mode (such as LM, LM-A or LM-L) of the current block;

k. kMax and/or F and/or S and/or *offset* may depend on the width and/or height of the current block.

l. kMax and/or F and/or S and/or *offset* may depend on H.

m. kMax and/or F and/or S and/or *offset* may depend on availability of neighbouring samples.

n. For example, kMax=1, F= H/4, S= H/2, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM coded, both the left and above neighbouring samples are available, and H>=4.

o. For example, kMax=3, F= H/8, S= H/4, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM coded, only above neighbouring samples are available, and H>=4.

p. For example, kMax=3, F= H/8, S= H/4, *offset* = 0. Alternatively, furthermore, the settings are done if the current block is LM-L coded and H>=4.

q. For example, kMax=1, F= 0, S= 1, *offset* = 0 if H is equal to 2.

[00263] Example 18: It is proposed two or four neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) are selected to derive the linear model parameters.

a. In one example, maxY/maxC and minY/minC are derived from two or four neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled), and are then used to derive the linear model parameters with the 2-point approach.

b. In one example, if there are two neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) are selected to derive the maxY/maxC and minY/minC, minY is set to be the smaller luma sample value and minC is its corresponding chroma sample value; maxY is set to be the larger luma sample value and maxC is its corresponding chroma sample value.

c. In one example, if there are four neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) are selected to derive the maxY/maxC and minY/minC, the luma samples and their corresponding chroma samples are divided into two arrayes G0 and G1, each contains two luma samples and their corresponding luma samples.

   i. Suppose the four luma samples and their corresponding chroma samples are denoted as S0, S1, S2, S3, then they can be divided into two groups in any order. For example:

      (a) G0={S0, S1}, G1={S2, S3};

      (b) G0={ S1, S0}, G1={S3, S2};

      (c) G0={S0, S2}, G1={S1, S3};

      (d) G0={S2, S0}, G1={S3, S1};

(e) G0={S1, S2}, G1={S0, S3};
(f) G0={S2, S1}, G1={S3, S0};
(g) G0={S0, S3}, G1={S1, S2};
(h) G0={S3, S0}, G1={S2, S1};
(i) G0={S1, S3}, G1={S0, S2};
(j) G0={S3, S1}, G1={S2, S0};
(k) G0={S3, S2}, G1={S0, S1};
(l) G0={S2, S3}, G1={S1, S0};
(m) G0 and G1 may be swaped.

ii. In one example, Luma sample value of G0[0] and G0[1] are compared, if luma sample value of G0[0] is larger than luma sample value of G0[1], the luma sample and its corresponding chroma sample of G0[0] are swapped with those onf G0[1].
    (a) Alternatively, if luma sample value of G0[0] is larger than or equal to luma sample value of G0[1], the luma sample and its corresponding chroma sample of G0[0] are swapped with those onf G0[1].
    (b) Alternatively, if luma sample value of G0[0] is smaller than luma sample value of G0[1], the luma sample and its corresponding chroma sample of G0[0] are swapped with those onf G0[1].
    (c) Alternatively, if luma sample value of G0[0] is smaller than or equal to luma sample value of G0[1], the luma sample and its corresponding chroma sample of G0[0] are swapped with those onf G0[1].

iii. In one example, Luma sample value of G1[0] and G1[1] are compared, if luma sample value of G1[0] is larger than luma sample value of G1[1], the luma sample and its corresponding chroma sample of G1[0] are swapped with those onf G1[1].
    (a) Alternatively, if luma sample value of G1[0] is larger than or equal to luma sample value of G1[1], the luma sample and its

corresponding chroma sample of G1[0] are swapped with those onf G1[1].

(b) Alternatively, if luma sample value of G1[0] is smaller than luma sample value of G1[1], the luma sample and its corresponding chroma sample of G1[0] are swapped with those onf G1[1].

(c) Alternatively, if luma sample value of G1[0] is smaller than or equal to luma sample value of G1[1], the luma sample and its corresponding chroma sample of G1[0] are swapped with those onf G1[1].

iv. In one example, Luma sample value of G0[0] and G1[1] are compared, if luma sample value of G0[0] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[1], then G0 and G1 are swapped.

(a) In one example, Luma sample value of G0[0] and G1[0] are compared, if luma sample value of G0[0] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[0], then G0 and G1 are swapped.

(b) In one example, Luma sample value of G0[1] and G1[0] are compared, if luma sample value of G0[1] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[0], then G0 and G1 are swapped.

(c) In one example, Luma sample value of G0[1] and G1[1] are compared, if luma sample value of G0[1] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[1], then G0 and G1 are swapped.

v. In one example, Luma sample value of G0[0] and G1[1] are compared, if luma sample value of G0[0] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[1], then G0[0] and G1[1] are swapped.

(a) In one example, Luma sample value of G0[0] and G1[0] are compared, if luma sample value of G0[0] is larger than (or smaller

than, or not larger than, or not smaller than) luma sample value of G1[0], then G0[0] and G1[0] are swapped.

(b) In one example, Luma sample value of G0[1] and G1[0] are compared, if luma sample value of G0[1] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[0], then G0[1] and G1[0] are swapped.

(c) In one example, Luma sample value of G0[1] and G1[1] are compared, if luma sample value of G0[1] is larger than (or smaller than, or not larger than, or not smaller than) luma sample value of G1[1], then G0[1] and G1[1] are swapped.

vi. In one example, maxY is calculated as the average of luma sample values of G0[0] and G0[1], maxC is calculated as the average of chroma sample values of G0[0] and G0[1].

(a) Alternatively, maxY is calculated as the average of luma sample values of G1[0] and G1[1], maxC is calculated as the average of chroma sample values of G1[0] and G1[1].

vii. In one example, minY is calculated as the average of luma sample values of G0[0] and G0[1], minC is calculated as the average of chroma sample values of G0[0] and G0[1].

Alternatively, minY is calculated as the average of luma sample values of G1[0] and G1[1], minC is calculated as the average of chroma sample values of G1[0] and G1[1].

d. In one example, if there are only two neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled) are available, they are first padded to be four chroma samples (and/or their corresponding luma samples), then the four chroma samples (and/or their corresponding luma samples) are used to derive the CCLM parameters.

i. In one example, two padding chroma samples (and/or their corresponding luma samples) are copied from the two available neighbouring chroma samples (and/or their corresponding luma samples which may be down-sampled)

[00264] Example 19: In all above examples, the selected chroma samples shall be located within the above row (i.e., with W samples) as depicted in FIG. 10, and/or the left column (i.e., with H samples) wherein W and H are the width and height of the current block.

a. Alternatively, above restriction may be applied when current block is coded with the normal LM mode.
b. Alternatively, the selected chroma samples shall be located within the above row (i.e., with W samples) and above-right row with H samples.
    i. Alternatively, furthermore, the above restriction may be applied when the current block is coded with the LM-A mode.
    ii. Alternatively, furthermore, the above restriction may be applied when the current block is coded with the LM-A mode or the normal LM mode with above row available but left column is unavailable.
c. Alternatively, the selected chroma samples shall be located within the left column (i.e., with H samples) and below-left column with W samples.
    i. Alternatively, furthermore, the above restriction may be applied when the current block is coded with the LM-L mode.
    ii. Alternatively, furthermore, the above restriction may be applied when the current block is coded with the LM-L mode or the normal LM mode with above row unavailable but left column is available.

[00265] Example 20

[00266] In one example, only the neighbouring luma samples at the positions where corresponding chroma samples are required to derive the CCLM parameters, need to be down-sampled.

[00267] Example 21

[00268] How to conduct the methods disclosed in this document may depend on the color format (such as 4:2:0 or 4:4:4).

a. Alternatively, how to conduct the methods disclosed in this document may depend on the bit-dpeth (such as 8-bit or 10-bit).
b. Alternatively, how to conduct the methods disclosed in this document may depend on the color representation method (such as RGB or YCbCr).

c. Alternatively, how to conduct the methods disclosed in this document may depend on the color representation method (such as RGB or YCbCr).
d. Alternatively, how to conduct the methods disclosed in this document may depend on the chroma down-sampling location.

[00269] Example 22

[00270] Whether to derive the maximum/minimum values of luma and chroma components used to derive CCLM parameters may depend on the availability of left and above neighbours. For example, the maximum/minimum values for luma and chroma components used to derive CCLM parameters may not be derived if both the left and above neighbouring blocks are unavailable.

a. Whether to derive the maximum/minimum values of luma and chroma components used to derive CCLM parameters may depend on the number of available neighbor samples. For example, the maximum/minimum values for luma and chroma components used to derive CCLM parameters may not be derived if numSampL == 0 and numSampT == 0. In another example, the maximum/minimum values for luma and chroma components used to derive CCLM parameters may not be derived if numSampL + numSampT == 0. In the two examples, numSampL and numSampT are the number of available neighbouring samples from left and above neighbouring blocks.
b. Whether to derive the maximum/minimum values of luma and chroma components used to derive CCLM parameters may depend on the number of picked samples used to derive the parameters. For example, the maximum/minimum values for luma and chroma components used to derive CCLM parameters may not be derived if cntL == 0 and cntT == 0. In another example, the maximum/minimum values for luma and chroma components used to derive CCLM parameters may not be derived if cntL + cntT == 0. In the two examples, cntL and cntT are the number of picked samples from left and above neighbouring blocks.

[00271] Example 23

[00272] In one example, the proposed method to derive the parameters used in CCLM, may be used to derive the parameters used in LIC or other coding tools that relies on linear model.

a. The bullets disclosed above may be applied to LIC, such as by replacing "chroma neighbouring samples" by "neighbouring samples of the current block" and replacing "corresponding luma samples" by "neighbouring samples of the reference block".
b. In one example, the samples utilized for LIC parameter derivation may exclude samples certain positions in the above row and/or left column.
    i. In one example, the samples utilized for LIC parameter derivation may exclude the first one in the above row.
        (a) Suppose that the top-left sample's coordinate is (x0, y0), it is proposed to exclude (x0, y0-1) for the usage of LIC parameters.
    ii. In one example, the samples utilized for LIC parameter derivation may exclude the first one in the left column.
        (a) Suppose that the top-left sample's coordinate is (x0, y0), it is proposed to exclude (x0-1, y0) for the usage of LIC parameters.
    iii. Whether to apply above methods and/or how to define the certain positions may depend on the availability of left column/above row.
    iv. Whether to apply above methods and/or how to define the certain positions may depend on block dimension.
c. In one example, N neighbouring samples (which may be down-sampled) of the current block and N corresponding neighbouring samples (which may be down-sampled correspondingly) of the reference block may be used to derive the parameters used for LIC.
    i. For example, N is 4.
    ii. In one example, the N neighboring samples may be defined as N/2 samples from above row; and N/2 samples from left column.
        (a) Alternatively, the N neighboring samples may be defined as N samples from above row or left column.
    iii. In another example, N is equal to min (L, T), where T is the total number of available neighbouring samples (which may be down-sampled) of the current block.
        (a) In one example, L is set to 4

iv. In one example, the selection of the coordinates of the N samples may follow the rule for selecting N samples in the CCLM process.
v. In one example, the selection of the coordinates of the N samples may follow the rule for selecting N samples in the LM-A process.
vi. In one example, the selection of the coordinates of the N samples may follow the rule for selecting N samples in the LM-L process.
vii. In one example, how to select the N samples may depend on the availability of above row/left column.

d. In one example, the N neighbouring samples (which may be down-sampled) of the current block and the N corresponding neighbouring samples (which may be down-sampled correspondingly) of the reference block are used to derive the parameters used in LIC, may be picked up based on sample positions.
i. The picking up method may depend on width and height of the current block.
ii. The picking up method may depend on the availability of the neighbouring blocks.
iii. For example, K1 neighbouring samples may be picked up from the left neighbouring samples and K2 neighbouring samples are picked up from the above neighbouring samples, if both above and left neighbouring samples are available. E.g. K1=K2=2.
iv. For example, K1 neighbouring samples may be picked up from the left neighbouring samples if only left neighbouring samples are available. E.g. K1 =4.
v. For example, K2 neighbouring samples may be picked up from the above neighbouring samples if only above neighbouring samples are available. E.g. K2 =4.
vi. For example, the above samples may be picked up with a first position offset value (denoted as F) and a step value (denoted as S) which may depend on the dimension of the current block and the availability of the neighbouring blocks.

(a) For example, methods disclosed in bullet 13 can be applied to derive F and S.

vii. For example, the left samples may be picked up with a first position offset value (denoted as F) and a step value (denoted as S) which may depend on the dimension of the current block and the availability of the neighboring blocks.

(a) For example, methods disclosed in bullet 14 can be applied to derive F and S.

e. In one example, the proposed method to derive the parameters used in CCLM, may also be used to derive the parameters used in LIC, when the current block is affine-coded.

f. The above methods may be used to derive the parameters used in other coding tools that relies on linear model.

[00273] In another example, cross-component prediction mode is proposed wherein the chroma samples are predicted with corresponding reconstructed luma samples according to the prediction model, as shown in Eq. 12. In Eq. 12, $Pred_C(x, y)$ denotes a prediction sample of chroma. α and β are two model parameters. $Rec'L(x, y)$ is a down-sampled luma sample.

$$Pred_C(x, y) = \alpha \times Rec'_L(x, y) + \beta, \tag{12}$$

[00274] A six-tap filter is introduced for the luma down-sampled process for block A in FIG. 11, as shown in Eq. 13.

$$\begin{aligned} Rec'_L(x, y) = (&2 \times Rec_L(2x, 2y) + 2 \times Rec_L(2x, 2y + 1) \\ &+ Rec_L(2x - 1, 2y) + Rec_L(2x + 1, 2y) \\ &+ Rec_L(2x - 1, 2y + 1) \\ &+ Rec_L(2x + 1, 2y + 1) + 4) >> 3. \end{aligned} \tag{13}$$

[00275] The above surrounding luma reference samples shaded in FIG. 11 are down-sampled with a 3-tap filter, as shown in Eq. 14. The left surrounding luma reference samples are down-sampled according to Eq. 15. If the left or above samples are not available, a 2-tap filter defined in Eq. 16 and Eq. 17 will be used.

$$Rec'_L(x, y) = \left(2 \times Rec_L(2x, 2y) + Rec_L(2x - 1, 2y) + Rec_L(2x + 1, 2y)\right) \gg 2 \tag{14}$$

$$Rec_L'(x,y) = (2 \times Rec_L(2x,2y) + Rec_L(2x,2y+1) + Rec_L(2x,2y-1)) \gg 2 \quad (15)$$

$$Rec_L'(x,y) = (3 \times Rec_L(2x,2y) + Rec_L(2x+1,2y) + 2) \gg 2 \quad (16)$$

$$Rec_L'(x,y) = (3 \times Rec_L(2x,2y) + Rec_L(2x,2y+1) + 2) \gg 2 \quad (17)$$

[00276] In particular, the surrounding luma reference samples are down sampled to the equal size to the chroma reference samples. The size is denoted as *width* and *height*. To derive α and β, only two or four neighboring samples are involved. A look-up table is applied to avoid the division operation when deriving α and β. The derivation methods is illustrated below.

3.1 Exemplary methods with up to two samples

[00277] (1) The ratio *r* of *width* and *height* is calculated as shown in Eq. 18.

$$r = \begin{cases} \frac{width}{height} & if\ width \geq height \\ \frac{height}{width} & if\ height < width \end{cases} \quad (18)$$

[00278] (2) If the above and the left blocks are both available, 2 samples locating at *posA* of the first above line, and *posL* of the first left line are selected. To simplify the description, width is assumed as the longer side. The derivation of *posA* and *posL* is shown in Eq. 19 (The position index starts from 0). FIG. 12 shows some examples of different width and height ratio (1, 2, 4 and 8, respectively). The selected samples are shaded.

$$posA = width - r$$
$$posL = height \quad (19)$$

[00279] (3) If the above block is available while the left block is not available, the first and the *posA* points of the above line are selected, as shown in FIG. 13.

[00280] (4) If the left block is available while the above block is not available, the first and the *posL* points of the left line are selected, as shown in FIG. 14.

[00281] (5) A chroma prediction model is derived according to the luminance and chrominance values of selected samples.

[00282] (6) If neither of the left and above blocks are available, a default prediction model is used, with α equals 0, β equals to 1<<(BitDepth-1), where BitDepth represents the bit-depth of chroma samples.

3.2 Exemplary methods with up to four samples

[00283] (1) The ratio *r* of *width* and *height* is calculated as Eq. 18.

[00284] (2) If the above and the left blocks are both available, 4 samples locating at the first and *posA* of the first above line, the first and *posL* of the first left line are selected. The derivation of *posA* and *posL* is illustrated in Eq. 19. FIG. 15 shows some examples of different width and height ratio (1, 2, 4 and 8, respectively). The selected samples are shaded.

[00285] (3) If the above block is available while the left block is not available, the first and the posA points of the above line are selected, as shown in FIG. 13.

[00286] (4) If the left block is available while the above block is not available, the first and the posL points of the left line are selected, as shown in FIG. 14.

[00287] (5) If neither of the left and above blocks are available, a default prediction model is used, with α equals 0, β equals to 1<<(BitDepth-1), where BitDepth represents the bit-depth of chroma samples.

3.3 Exemplary methods that use lookup tables in LM derivation

[00288] FIG. 16 shows an example of lookup tables with 128, 64 and 32 entries and each entry is represented by 16 bits. The 2-point LM derivation process is simplified as shown in Table 1 and FIG. 17 with 64 entries. It should be noted that the first entry may not be stored into the table.

[00289] It should also be noted that although each entry in the exemplary tables are designed to be with 16 bits, it can be easily transformed to a number with less bits (such as 8 bits or 12 bits). For example, a table of entries with 8 bits can be attained as:

[00290] g_aiLMDivTableHighSimp_64_8[i] = (g_aiLMDivTableHighSimp_64[i]+128) >> 8.

[00291] For example, a table of entries with 12 bits can be attained as:

[00292] g_aiLMDivTableHighSimp_64_12[i] = (g_aiLMDivTableHighSimp_64[i]+8) >> 4.

Table 1: Simplified LM derivation process

```
int iDeltaLuma = maxLuma – minLuma;
const int TABLE_PRECISION = 16; // It may be 8 or 12.
const int BIT_DEPTH = 10; // Bit depth for samples.
int shift = TABLE_PRECISION;
if( iDeltaLuma > 64) {
        int depthshift = BIT_DEPTH - 6; // 64 is equal to 2^6.
    iDeltaLuma = ( iDeltaLuma + (1<<(depthshift-1)))>> depthshift;
```

```
    shift -= depthshift;
}
a = (((maxChroma – minChroma)*g_aiLMDivTableHighSimp_64[iDeltaLuma-1] + (1<<(shift-1)))>>shift;
```

[00293] It should be noted that maxLuma and minLuma may indicate the maximum and minimum luma samples values of selected positions. Alternatively, they may indicate a function of maximum and minimum luma samples values of selected positions, such as averaging. When there are only 4 positions selected, they may also indicate the average of two larger luma values and average of two smaller luma values. Further note that in FIG. 17, maxChroma and minChroma represents the chroma values corresponding to maxLuma and minLuma.

[00294] **3.3 Method #4 with up to four samples**

[00295] Suppose the block width and height of current chroma block is W and H, respectively. And the top-left coordinate of current chroma block is [0, 0].

[00296] If the above and the left blocks are both available and current mode is the normal LM mode (excluding LM-A, and LM-L), 2 chroma samples locating at above row, and 2 chroma samples located left column are selected.

[00297] The two above samples' coordinates are [floor(W/4), -1] and [floor (3*W/4), -1].

[00298] The two left samples' coordinates are [-1, floor(H/4)] and [-1, floor (3*H/4)].

[00299] The selected samples are painted in red as depicted in FIG. 31A.

[00300] Subsequently, the 4 samples are sorted according to luma sample intensity and classified into 2 group. The two larger samples and two smaller samples are respectively averaged. Cross component prediction model is derived with the 2 averaged points. Alternatively, the maximum and minimum value of the four samples are used to derive the LM parameters.

[00301] If the above block is available while the left block is not available, four chroma samples from above block are selected when W>2 and 2 chroma samples are selected when W=2.

[00302] The four selected above samples' coordinates are [W/8, -1], [W/8 + W/4, -1], [W/8 + 2*W/4, -1], and [W/8 + 3*W/4 , -1].

[00303] The selected samples are painted in red as depicted in FIG. 31B

[00304] If the left block is available while the above block is not available, four chroma samples from left block are selected when H>2 and 2 chroma samples are selected when H=2.

[00305] The four selected left samples' coordinates are [-1, H/8], [-1, H/8 + H/4], [-1, H/8 +

2*H/4, -1], and [-1, H/8 + 3*H/4].

[00306] If neither of the left and above blocks are available, a default prediction is used. with α equals 0, β equals to 1<<(BitDepth-1), where BitDepth represents the bit-depth of chroma samples.

[00307] If the current mode is the LM-A mode, four chroma samples from above block are selected when W'>2 and 2 chroma samples are selected when W'=2. W' is the available number of above neighbouring samples, which can be 2*W.

[00308] The four selected above samples' coordinates are [W'/8, -1], [W'/8 + W'/4, -1], [W'/8 + 2*W'/4, -1], and [W'/8 + 3*W'/4 , -1].

[00309] If the current mode is the LM-L mode, four chroma samples from left block are selected when H'>2 and 2 chroma samples are selected when H'=2. H' is the available number of left neighbouring samples, which can be 2*H.

[00310] The four selected left samples' coordinates are [-1, H'/8], [-1, H'/8 + H'/4], [-1, H'/8 + 2*H'/4, -1], and [-1, H'/8 + 3*H'/4].

[00311] 3.5 Example embodiment for modifying current VVC standard for use of CCLM prediction.

[00312] ***8.3.4.2.8 Specification of INTRA_LT_CCLM, INTRA_L_CCLM and INTRA_T_CCLM intra prediction mode***

[00313] The equations are described in this section using the equation numbers corresponding to those in the current draft of the VVC standard.

[00314] Inputs to this process are:

- the intra prediction mode predModeIntra,
- a sample location (xTbC, yTbC) of the top-left sample of the current transform block relative to the top-left sample of the current picture,
- a variable nTbW specifying the transform block width,
- a variable nTbH specifying the transform block height,
- chroma neighbouring samples p[x][y], with x = −1, y = 0..2 * nTbH − 1 and x = 0.. 2 * nTbW − 1, y = − 1.

[00315] Output of this process are predicted samples predSamples[x][y], with x = 0..nTbW − 1, y = 0..nTbH − 1.

[00316] The current luma location (xTbY, yTbY) is derived as follows:

$$(xTbY, yTbY) = (xTbC << 1, yTbC << 1) \quad (8\text{-}155)$$

[00317] The variables availL, availT and availTL are derived as follows:

…

– If predModeIntra is equal to INTRA_LT_CCLM, the following applies:

numSampT = availT ? nTbW : 0 (8-156)

numSampL = availL ? nTbH : 0 (8-157)

– Otherwise, the following applies:

numSampT = (availT && predModeIntra == INTRA_T_CCLM) ? (nTbW + numTopRight) : 0 (8-158)

numSampL = (availL && predModeIntra == INTRA_L_CCLM) ? (nTbH + numLeftBelow) : 0 (8-159)

[00318] The variable bCTUboundary is derived as follows:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) == 0) ? TRUE : FALSE. (8-160)

[00319] The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:

– If both numSampL and numSampT are equal to 0, the following applies:

predSamples[x][y] = 1 << ($BitDepth_C$ − 1) (8-161)

– Otherwise, the following ordered steps apply:

1. …[no changes to current specification]
2. …
3. …
4. …
5. …
6. ……[no changes to current specification]
7. The variables minY, maxY, minC and maxC are derived as follows:
   - The variable minY is set equal to 1 << ($BitDepth_Y$) + 1 and the variable maxY is set equal to −1.
   - If availL is equal to TRUE and predModeIntra is equal to INTRA_LT_CCLM, the variable aboveIs4 is set equal to 0;Otherwise, it is set equal to 1.
   - If availT is equal to TRUE and predModeIntra is equal to INTRA_LT_CCLM, the variable LeftIs4 is set equal to 0;Otherwise, it is set equal to 1.
   - The variable arrays startPos[] and pickStep[] are derived as follows:

- startPos[0] = actualTopTemplateSampNum >> (2 + aboveIs4);
- pickStep[0] = std::max(1, actualTopTemplateSampNum >> (1 + aboveIs4));
- startPos[1] = actualLeftTemplateSampNum >> (2 + leftIs4);
- pickStep[1] = std::max(1, actualLeftTemplateSampNum >> (1 + leftIs4));

- The variable cnt is set equal to 0.
- If predModeIntra is equal to INTRA_LT_CCLM, the variable nSX is set equal to nTbW, nSY is set equal to nTbH; Otherwise, nSX is set equal to numSampLT and nSY is set equal to numSampL.
- If availT is equal to TRUE and predModeIntra is not equal to INTRA_L_CCLM, the variables selectLumaPix, selectChromaPix are derived as follows:
  - While startPos[0]+cnt* pickStep[0] < nSX and cnt < 4, the following applies:
    - selectLumaPix[cnt] = pTopDsY[startPos[0]+cnt* pickStep[0]];
    - selectChromaPix[cnt]= p[startPos[0]+cnt* pickStep[0]][−1];
    - cnt++;
- If availL is equal to TRUE and predModeIntra is not equal to INTRA_T_CCLM, the variables selectLumaPix, selectChromaPix are derived as follows:
  - While startPos[1]+cnt* pickStep[1] < nSY and cnt < 4, the following applies:
    - selectLumaPix[cnt] = pLeftDsY [startPos[1]+cnt* pickStep[1]];
    - selectChromaPix[cnt]= p[-1][startPos[1]+cnt* pickStep[1]];
    - cnt++;
- If cnt is equal to 2, the following applies:
  - If selectLumaPix[0] > selectLumaPix[1], minY is set equal to selectLumaPix[1], minC is set equal to selectChromaPix[1], maxY is set equal to selectLumaPix[0] and maxC is set equal to selectChromaPix[0]; Otherwise, maxY is set equal to selectLumaPix[1], maxC is set equal to selectChromaPix[1], minY is set equal to selectLumaPix[0] and minC is set equal to selectChromaPix[0]
- Otherwise, if cnt is equal to 4, the following applies:
  - The variable arrays minGrpIdx and maxGrpIdx are initialized as:
    - minGrpIdx[0] = 0, minGrpIdx[1] = 1, maxGrpIdx[0] = 2, maxGrpIdx[1] = 3;
  - The following applies
    - If selectLumaPix[minGrpIdx[0]] > selectLumaPix[minGrpIdx[1]], swap minGrpIdx[0] and minGrpIdx[1];
    - If selectLumaPix[maxGrpIdx[0]] > selectLumaPix[maxGrpIdx[1]], swap maxGrpIdx[0] and maxGrpIdx[1];

- If selectLumaPix[minGrpIdx[0]] > selectLumaPix[maxGrpIdx[1]], swap minGrpIdx and maxGrpIdx;
- If selectLumaPix[minGrpIdx[1]] > selectLumaPix[maxGrpIdx[0]], swap minGrpIdx[1] and maxGrpIdx[0];

– maxY, maxC, minY and minC are derived as follows:

- maxY =(selectLumaPix[maxGrpIdx[0]]+selectLumaPix[maxGrpIdx[1]]+1) >>1;
- maxC =(selectChromaPix[maxGrpIdx[0]]+selectChromaPix[maxGrpIdx[1]] +1)

  >>1;
- maxY =(selectLumaPix[minGrpIdx [0]]+selectLumaPix[minGrpIdx [1]]+1)>>1;
- maxC =(selectChromaPix[minGrpIdx[0]]+ selectChromaPix [minGrpIdx [1]]+1)>>1;
-

8. The variables a, b, and k are derived as follows:

[00320] [end of changes]

[00321] **3.6 Another exemplary working draft on proposed CCLM prediction**

[00322] In this section, another exemplary embodiment that shows modifications that can be made to the current working draft of the VVC standard is described. The equation numbers here refer to the corresponding equation numbers in the VVC standard.

[00323] Specification of INTRA_LT_CCLM, INTRA_L_CCLM and INTRA_T_CCLM intra prediction mode.

[00324] [add to the current VVC working draft as below]

The number of available neighbouring chroma samples on the top and top-right numTopSamp and the number of available neighbouring chroma samples on the left and left-below nLeftSamp are derived as follows:

– If predModeIntra is equal to INTRA_LT_CCLM, the following applies:

numSampT = availT ? nTbW : 0 (8-157)

numSampL = availL ? nTbH : 0 (8-158)

– Otherwise, the following applies:

numSampT = (availT && predModeIntra == INTRA_T_CCLM) ?

(nTbW + Min(numTopRight, nTbH)) : 0 (8-159)

numSampL = (availL && predModeIntra = = INTRA_L_CCLM) ?

(nTbH + Min(numLeftBelow, nTbW)) : 0 (8-160)

The variable bCTUboundary is derived as follows:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ? TRUE : FALSE. (8-161)

The variable cntN and array pickPosN[] with N being replaced by L and T, are derived as follows:

- The variable numIs4N is set equal to ((availN && predModeIntra == INTRA_LT_CCLM) ? 0 : 1).
- The variable startPosN is set equal to numSampN >> (2 + numIs4N).
- The variable pickStepN is set equal to Max(1, numSampN >> (1 + numIs4N)).
- If availN is equal to TRUE and predModeIntra is equal to INTRA_LT_CCLM or INTRA_N_CCLM, cntN is set equal to (1 + numIs4N) << 1, and pickPosN[pos] is set equal to (startPosN + pos * pickStepN), with pos = 0..(cntN – 1).
- Otherwise, cntN is set equal to 0.

The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:

- If both numSampL and numSampT are equal to 0, the following applies:

  predSamples[x][y] = 1 << ($BitDepth_C$ − 1) (8-162)

- Otherwise, the following ordered steps apply:
  1. The collocated luma samples pY[x][y] with x = 0..nTbW * 2 − 1, y= 0..nTbH * 2 − 1 are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x, yTbY + y).
  2. The neighbouring luma samples samples pY[x][y] are derived as follows:
     - When numSampL is greater than 0, the neighbouring left luma samples pY[x][y] with x = −1..−3, y = 0..2 * numSampL − 1, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x , yTbY +y).
     - When numSampT is greater than 0, the neighbouring top luma samples pY[x][y] with x = 0..2 * numSampT − 1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
     - When availTL is equal to TRUE, the neighbouring top-left luma samples pY[x][y] with x = −1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
  3. The down-sampled collocated luma samples pDsY[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:
     - If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– pDsY[x][y] with x = 1..nTbW − 1, y = 1..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2 * x][2 * y - 1] + pY[2 * x - 1][2 * y] + 4 * pY[2 * x][2 * y] + pY[2 * x + 1][2 * y] + pY[2 * x][2 * y + 1] + 4) >> 3 \quad (8\text{-}163)$$

– If availL is equal to TRUE, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2 * y - 1] + pY[-1][2 * y] + 4 * pY[0][2 * y] + pY[1][2 * y] + pY[0][2 * y + 1] + 4) >> 3 \quad (8\text{-}164)$$

– Otherwise, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2 * y - 1] + 2 * pY[0][2 * y] + pY[0][2 * y + 1] + 2) >> 2 \quad (8\text{-}165)$$

– If availT is equal to TRUE, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2 * x][-1] + pY[2 * x - 1][0] + 4 * pY[2 * x][0] + pY[2 * x + 1][0] + pY[2 * x][1] + 4) >> 3 \quad (8\text{-}166)$$

– Otherwise, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2 * x - 1][0] + 2 * pY[2 * x][0] + pY[2 * x + 1][0] + 2) >> 2 \quad (8\text{-}167)$$

– If availL is equal to TRUE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + pY[-1][0] + 4 * pY[0][0] + pY[1][0] + pY[0][1] + 4) >> 3 \quad (8\text{-}168)$$

– Otherwise if availL is equal to TRUE and availT is equal to FALSE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[-1][0] + 2 * pY[0][0] + pY[1][0] + 2) >> 2 \quad (8\text{-}169)$$

– Otherwise if availL is equal to FALSE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + 2 * pY[0][0] + pY[0][1] + 2) >> 2 \quad (8\text{-}170)$$

– Otherwise (availL is equal to FALSE and availT is equal to FALSE), pDsY[0][0] is derived as follows:

$$pDsY[0][0] = pY[0][0] \quad (8\text{-}171)$$

– Otherwise, the following applies:

– pDsY[x][y] with x = 1..nTbW − 1, y = 0..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2*x-1][2*y] + pY[2*x-1][2*y+1] + 2*pY[2*x][2*y] + 2*pY[2*x][2*y+1] + pY[2*x+1][2*y] + pY[2*x+1][2*y+1] + 4) >> 3 \quad (8\text{-}172)$$

– If availL is equal to TRUE, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[-1][2*y] + pY[-1][2*y+1] + 2*pY[0][2*y] + 2*pY[0][2*y+1] + pY[1][2*y] + pY[1][2*y+1] + 4) >> 3 \quad (8\text{-}173)$$

– Otherwise, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2*y] + pY[0][2*y+1] + 1) >> 1 \quad (8\text{-}174)$$

4. When numSampL is greater than 0, the selcted neighbouring left chroma samples pSelC[idx] are set equal to p[-1][pickPosL[idx]] with idx = 0..(cntL – 1), and the selected down-sampled neighbouring left luma samples pSelDsY[idx] with idx = 0..(cntL-1) are derived as follows:

– The variable y is set equal to pickPosL[idx].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If y > 0 || availTL == TRUE,

$$pSelDsY[idx] = (pY[-2][2*y-1] + pY[-3][2*y] + 4*pY[-2][2*y] + pY[-1][2*y] + pY[-2][2*y+1] + 4) >> 3 \quad (8\text{-}175)$$

– Otherwise,

$$pSelDsY[idx] = (pY[-3][0] + 2*pY[-2][0] + pY[-1][0] + 2) >> 2 \quad (8\text{-}177)$$

– Otherwise, the following applies:

$$pSelDsY[idx] = (pY[-1][2*y] + pY[-1][2*y+1] + 2*pY[-2][2*y] + 2*pY[-2][2*y+1] + pY[-3][2*y] + pY[-3][2*y+1] + 4) >> 3 \quad (8\text{-}178)$$

5. When numSampT is greater than 0, the selcted neighbouring top chroma samples pSelC[idx] are set equal to p[pickPosT[idx]][-1] with idx = 0..(cntT – 1), and the down-sampled neighbouring top luma samples pSelDsY[idx] with idx = cntL..(cntL + cntT – 1) are specified as follows:

– The variable x is set equal to pickPosT[idx - cntL].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If x > 0:

– If bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[2 * x][-3] + pY[2 * x - 1][-2] + 4 * pY[2 * x][-2] + pY[2 * x + 1][-2] + pY[2 * x][-1] + 4) >> 3 \quad (8\text{-}179)$$

– Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-1] + 2* pY[2 * x][-1] + pY[2 * x + 1][-1] + 2) >> 2 \quad (8\text{-}180)$$

– Otherwise:

– If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + pY[-1][-2] + 4 * pY[0][-2] + pY[1][-2] + pY[0][-1] + 4) >> 3 \quad (8\text{-}181)$$

– Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$pSelDsY[idx] = (pY[-1][-1] + 2* pY[0][-1] + pY[1][-1] + 2) >> 2 \quad (8\text{-}182)$$

– Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + 2 * pY[0][-2] + pY[0][-1] + 2) >> 2 \quad (8\text{-}183)$$

– Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}184)$$

– Otherwise, the following applies:

– If x > 0:

– If bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-2] + pY[2 * x - 1][-1] + 2* pY[2 * x][-2] + 2*pY[2 * x][-1] + pY[2 * x + 1][-2] + pY[2 * x + 1][-1] + 4) >> 3 \quad (8\text{-}185)$$

– Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-1] + 2* pY[2 * x][-1] + pY[2 * x + 1][-1] + 2) >> 2 \quad (8\text{-}186)$$

– Otherwise:

– If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[-1][-2] + pY[-1][-1] + 2* pY[0][-2] + 2*pY[0][-1] + pY[1][-2] + pY[1][-1] + 4) >> 3 \quad (8\text{-}187)$$

– Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$pSelDsY[idx] = (pY[-1][-1] + 2* pY[0][-1] + pY[1][-1] + 2) >> 2 \quad (8\text{-}188)$$

– Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-2] + pY[0][-1] + 1) >> 1 \quad (8\text{-}189)$$

– Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}190)$$

6. The variables minY, maxY, minC and maxC are derived as follows:
   –
   – When cntT+cntL is equal to 2, set pSelC[idx + 2] = pSelC[idx] and pSelDsY[idx + 2] = pSelDsY[idx], with idx = 0 and 1.
   – The arrays minGrpIdx[] and maxGrpIdx[] are set as: minGrpIdx[0] = 0, minGrpIdx[1] = 1, maxGrpIdx[0] = 2, maxGrpIdx[1] = 3.
   – If pSelDsY[minGrpIdx[0]] > pSelDsY[minGrpIdx[1]], Swap(minGrpIdx[0], minGrpIdx[1]).
   – If pSelDsY[maxGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(maxGrpIdx[0], maxGrpIdx[1]).
   – If pSelDsY[minGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(minGrpIdx, maxGrpIdx).
   – If pSelDsY[minGrpIdx[1]] > pSelDsY[maxGrpIdx[0]], Swap(minGrpIdx[1], maxGrpIdx[0]).
   – maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1.
   – maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1.
   – minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1.
   – minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1.
7. The variables a, b, and k are derived as follows:

– If numSampL is equal to 0, and numSampT is equal to 0, the following applies:

$$k = 0 \quad (8\text{-}208)$$

$$a = 0 \quad (8\text{-}209)$$

$$b = 1 << (BitDepth_C - 1) \quad (8\text{-}210)$$

– Otherwise, the following applies:

$$diff = maxY - minY \quad (8\text{-}211)$$

– If diff is not equal to 0, the following applies:

$$diffC = maxC - minC \quad (8\text{-}212)$$

$$x = Floor(Log2(diff)) \quad (8\text{-}213)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (8\text{-}214)$$

$$x += (normDiff \ != \ 0) \ ? \ 1 \ : \ 0 \quad (8\text{-}215)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (8\text{-}216)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (8\text{-}217)$$

$$k = ((3 + x - y) < 1) \ ? \ 1 \ : \ 3 + x - y \quad (8\text{-}218)$$

$$a = ((3 + x - y) < 1) \ ? \ Sign(a) * 15 \ : \ a \quad (8\text{-}219)$$

$$b = minC - ((a * minY) >> k) \quad (8\text{-}220)$$

where divSigTable[ ] is specified as follows:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (8\text{-}221)$$

– Otherwise (diff is equal to 0), the following applies:

$$k = 0 \quad (8\text{-}222)$$

$$a = 0 \quad (8\text{-}223)$$

$$b = minC \quad (8\text{-}224)$$

8. The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0.. nTbH − 1 are derived as follows:

$$predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b) \quad (8\text{-}225)$$

[00325] [End of the embodiment example]

[00326] **3.7 Another exemplary working draft on proposed CCLM prediction**

[00327] In this section, another exemplary embodiment that shows modifications that can be

made to the current working draft of the VVC standard is described. The equation numbers here refer to the corresponding equation numbers in the VVC standard.

[00328] Specification of INTRA_LT_CCLM, INTRA_L_CCLM and INTRA_T_CCLM intra prediction mode

...

The number of available neighbouring chroma samples on the top and top-right numTopSamp and the number of available neighbouring chroma samples on the left and left-below nLeftSamp are derived as follows:

– If predModeIntra is equal to INTRA_LT_CCLM, the following applies:

numSampT = availT ? nTbW : 0 (8-157)

numSampL = availL ? nTbH : 0 (8-158)

– Otherwise, the following applies:

numSampT = (availT && predModeIntra == INTRA_T_CCLM) ?

(nTbW + Min(numTopRight, nTbH)) : 0 (8-159)

numSampL = (availL && predModeIntra == INTRA_L_CCLM) ?

(nTbH + Min(numLeftBelow, nTbW)) : 0 (8-160)

The variable bCTUboundary is derived as follows:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) == 0) ? TRUE : FALSE. (8-161)

The variable cntN and array pickPosN[] with N being replaced by L and T, are derived as follows:

– The variable numIs4N is set equal to ((availN && predModeIntra == INTRA_LT_CCLM) ? 0 : 1).

– The variable startPosN is set equal to numSampN >> (2 + numIs4N).

– The variable pickStepN is set equal to Max(1, numSampN >> (1 + numIs4N)).

– If availN is equal to TRUE and predModeIntra is equal to INTRA_LT_CCLM or INTRA_N_CCLM, cntN is set equal to Min(numSampN, (1 + numIs4N) << 1), and pickPosN[pos] is set equal to (startPosN + pos * pickStepN), with pos = 0..(cntN – 1).

– Otherwise, cntN is set equal to 0.

The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:

– If both numSampL and numSampT are equal to 0, the following applies:

predSamples[x][y] = 1 << ($BitDepth_C$ − 1) (8-162)

– Otherwise, the following ordered steps apply:

1. The collocated luma samples pY[x][y] with x = 0..nTbW * 2 − 1, y= 0..nTbH * 2 − 1 are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x, yTbY + y).
2. The neighbouring luma samples samples pY[x][y] are derived as follows:
   – When numSampL is greater than 0, the neighbouring left luma samples pY[x][y] with x = −1..−3, y = 0..2 * numSampL − 1, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x , yTbY +y).
   – When numSampT is greater than 0, the neighbouring top luma samples pY[x][y] with x = 0..2 * numSampT − 1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
   – When availTL is equal to TRUE, the neighbouring top-left luma samples pY[x][y] with x = −1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
3. The down-sampled collocated luma samples pDsY[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:
   – If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:
     – pDsY[x][y] with x = 1..nTbW − 1, y = 1..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2 * x][2 * y - 1] + pY[2 * x - 1][2 * y] + 4 * pY[2 * x][2 * y] + pY[2 * x + 1][2 * y] + pY[2 * x][2 * y + 1] + 4) >> 3 \quad (8\text{-}163)$$

     – If availL is equal to TRUE, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2 * y - 1] + pY[-1][2 * y] + 4 * pY[0][2 * y] + pY[1][2 * y] + pY[0][2 * y + 1] + 4) >> 3 \quad (8\text{-}164)$$

     – Otherwise, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2 * y - 1] + 2 * pY[0][2 * y] + pY[0][2 * y + 1] + 2) >> 2 \quad (8\text{-}165)$$

     – If availT is equal to TRUE, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2 * x][-1] + pY[2 * x - 1][0] + 4 * pY[2 * x][0] + pY[2 * x + 1][0] + pY[2 * x][1] + 4) >> 3 \quad (8\text{-}166)$$

     – Otherwise, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2 * x - 1][0] + 2 * pY[2 * x][0] + pY[2 * x + 1][0] + 2) >> 2 \quad (8\text{-}167)$$

– If availL is equal to TRUE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + pY[-1][0] + 4 * pY[0][0] + pY[1][0] + pY[0][1] + 4) >> 3 \quad (8\text{-}168)$$

– Otherwise if availL is equal to TRUE and availT is equal to FALSE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[-1][0] + 2 * pY[0][0] + pY[1][0] + 2) >> 2 \quad (8\text{-}169)$$

– Otherwise if availL is equal to FALSE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + 2 * pY[0][0] + pY[0][1] + 2) >> 2 \quad (8\text{-}170)$$

– Otherwise (availL is equal to FALSE and availT is equal to FALSE), pDsY[0][0] is derived as follows:

$$pDsY[0][0] = pY[0][0] \quad (8\text{-}171)$$

– Otherwise, the following applies:

– pDsY[x][y] with x = 1..nTbW − 1, y = 0..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2 * x - 1][2 * y] + pY[2 * x - 1][2 * y + 1] + 2* pY[2 * x][2 * y] + 2*pY[2 * x][2 * y + 1] + pY[2 * x + 1][2 * y] + pY[2 * x + 1][2 * y + 1] + 4) >> 3 \quad (8\text{-}172)$$

– If availL is equal to TRUE, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[-1][2 * y] + pY[-1][2 * y + 1] + 2* pY[0][2 * y] + 2*pY[0][2*y + 1] + pY[1][2 * y] + pY[1][2 * y + 1] + 4) >> 3 \quad (8\text{-}173)$$

– Otherwise, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2 * y] + pY[0][2 * y + 1] + 1) >> 1 \quad (8\text{-}174)$$

4. When numSampL is greater than 0, the selcted neighbouring left chroma samples pSelC[idx] are set equal to p[-1][pickPosL[idx]] with idx = 0..(cntL – 1), and the selected down-sampled neighbouring left luma samples pSelDsY[idx] with idx = 0..(cntL-1) are derived as follows:

– The variable y is set equal to pickPosL[idx].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If y > 0 || availTL == TRUE,

$$pSelDsY[idx] = (pY[-2][2 * y - 1] + pY[-3][2 * y] + 4 * pY[-2][2 * y] + pY[-1][2 * y] + pY[-2][2 * y + 1] + 4) >> 3 \quad (8\text{-}175)$$

– Otherwise,

$$pSelDsY[idx] = (pY[-3][0] + 2 * pY[-2][0] + pY[-1][0] + 2) >> 2 \quad (8\text{-}177)$$

– Otherwise, the following applies:

$$pSelDsY[idx] = (pY[-1][2 * y] + pY[-1][2 * y + 1] + 2* pY[-2][2 * y] + 2*pY[-2][2 * y + 1] + pY[-3][2 * y] + pY[-3][2 * y + 1] + 4) >> 3 \quad (8\text{-}178)$$

5. When numSampT is greater than 0, the selcted neighbouring top chroma samples pSelC[idx] are set equal to p[pickPosT[idx]][-1] with idx = 0..(cntT – 1), and the down-sampled neighbouring top luma samples pSelDsY[idx] with idx = cntL..(cntL + cntT – 1) are specified as follows:

– The variable x is set equal to pickPosT[idx - cntL].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If x > 0:

– If bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[2 * x][-3] + pY[2 * x - 1][-2] + 4 * pY[2 * x][-2] + pY[2 * x + 1][-2] + pY[2 * x][-1] + 4) >> 3 \quad (8\text{-}179)$$

– Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-1] + 2* pY[2 * x][-1] + pY[2 * x + 1][-1] + 2) >> 2 \quad (8\text{-}180)$$

– Otherwise:

– If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + pY[-1][-2] + 4 * pY[0][-2] + pY[1][-2] + pY[0][-1] + 4) >> 3 \quad (8\text{-}181)$$

– Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$pSelDsY[idx] = (pY[-1][-1] + 2* pY[0][-1] + pY[1][-1] + 2) >> 2 \quad (8\text{-}182)$$

- Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + 2 * pY[0][-2] + pY[0][-1] + 2) >> 2 \quad (8\text{-}183)$$

- Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}184)$$

- Otherwise, the following applies:
  - If x > 0:
    - If bCTUboundary is equal to FALSE, the following applies:

$$\begin{aligned} pSelDsY[idx] = (& pY[2 * x - 1][-2] + pY[2 * x - 1][-1] + \\ & 2* pY[2 * x][-2] + \quad 2*pY[2 * x][-1] + \\ & pY[2 * x + 1][-2] + pY[2 * x + 1][-1] + 4) >> 3 \end{aligned} \quad (8\text{-}185)$$

    - Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$\begin{aligned} pSelDsY[idx] = (& pY[2 * x - 1][-1] + \\ & 2* pY[2 * x][-1] + \\ & pY[2 * x + 1][-1] + 2) >> 2 \end{aligned} \quad (8\text{-}186)$$

  - Otherwise:
    - If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$\begin{aligned} pSelDsY[idx] = (& pY[- 1][-2] + pY[- 1][-1] + \\ & 2* pY[0][-2] + 2*pY[0][-1] + \\ & pY[1][-2] + pY[1][-1] + 4) >> 3 \end{aligned} \quad (8\text{-}187)$$

    - Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$\begin{aligned} pSelDsY[idx] = (& pY[- 1][-1] + \\ & 2* pY[0][-1] + \\ & pY[1][-1] + 2) >> 2 \end{aligned} \quad (8\text{-}188)$$

    - Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-2] + pY[0][-1] + 1) >> 1 \quad (8\text{-}189)$$

    - Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}190)$$

6. When cntT+ cntL is not equal to 0, the variables minY, maxY, minC and maxC are derived as follows:
   –
   – When cntT+cntL is equal to 2, set pSelComp[3] equal to pSelComp [0], pSelComp[2] equal to pSelComp[1], pSelComp[0] equal to pSelComp [1], and pSelComp[1] equal to pSelComp[3], with Comp being replaced by DsY and C.
   – The arrays minGrpIdx[] and maxGrpIdx[] are set as: minGrpIdx[0] = 0, minGrpIdx[1] = 1, maxGrpIdx[0] = 2, maxGrpIdx[1] = 3.
   – If pSelDsY[minGrpIdx[0]] > pSelDsY[minGrpIdx[1]], Swap(minGrpIdx[0], minGrpIdx[1]).
   – If pSelDsY[maxGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(maxGrpIdx[0], maxGrpIdx[1]).
   – If pSelDsY[minGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(minGrpIdx, maxGrpIdx).
   – If pSelDsY[minGrpIdx[1]] > pSelDsY[maxGrpIdx[0]], Swap(minGrpIdx[1], maxGrpIdx[0]).
   – maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1.
   – maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1.
   – minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1.
   – minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1.
7. The variables a, b, and k are derived as follows:
   – If numSampL is equal to 0, and numSampT is equal to 0, the following applies:

$$k = 0 \quad (8\text{-}208)$$

$$a = 0 \quad (8\text{-}209)$$

$$b = 1 << (BitDepth_C - 1) \quad (8\text{-}210)$$

   – Otherwise, the following applies:

$$diff = maxY - minY \quad (8\text{-}211)$$

   – If diff is not equal to 0, the following applies:

$$diffC = maxC - minC \quad (8\text{-}212)$$

$$x = Floor(Log2(diff)) \quad (8\text{-}213)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (8\text{-}214)$$

$$x += (normDiff \ != \ 0) \ ? \ 1 \ : \ 0 \quad (8\text{-}215)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (8\text{-}216)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (8\text{-}217)$$

$$k = ((3 + x - y) < 1) \ ? \ 1 \ : \ 3 + x - y \quad (8\text{-}218)$$

$$a = ((3 + x - y) < 1) \ ? \ Sign(a) * 15 \ : \ a \quad (8\text{-}219)$$

$$b = minC - ((a * minY) >> k) \quad (8\text{-}220)$$

where divSigTable[ ] is specified as follows:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (8\text{-}221)$$

– Otherwise (diff is equal to 0), the following applies:

$$k = 0 \quad (8\text{-}222)$$

$$a = 0 \quad (8\text{-}223)$$

$$b = minC \quad (8\text{-}224)$$

8. The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0.. nTbH − 1 are derived as follows:

$$predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b) \quad (8\text{-}225)$$

[00329] 3.8 An alternative working draft on proposed CCLM prediction

[00330] In this section, an alternative exemplary embodiment that shows another modifications that can be made to the current working draft of the VVC standard is described. The equation numbers here refer to the corresponding equation numbers in the VVC standard.

[00331] Specification of INTRA_LT_CCLM, INTRA_L_CCLM and INTRA_T_CCLM intra prediction mode.

...

The number of available neighbouring chroma samples on the top and top-right numTopSamp and the number of available neighbouring chroma samples on the left and left-below nLeftSamp are derived as follows:

– If predModeIntra is equal to INTRA_LT_CCLM, the following applies:

$$numSampT = availT \ ? \ nTbW \ : \ 0 \quad (8\text{-}157)$$

$$numSampL = availL \ ? \ nTbH \ : \ 0 \quad (8\text{-}158)$$

– Otherwise, the following applies:

$$numSampT = (availT \ \&\& \ predModeIntra \ == \ INTRA_T_CCLM) \ ?$$

(nTbW + Min(numTopRight, nTbH)) : 0 (8-159)

numSampL = (availL && predModeIntra = = INTRA_L_CCLM) ?

(nTbH + Min(numLeftBelow, nTbW)) : 0 (8-160)

The variable bCTUboundary is derived as follows:

bCTUboundary = (yTbC & (1 << (CtbLog2SizeY − 1) − 1) = = 0) ? TRUE : FALSE. (8-161)

The variable cntN and array pickPosN[] with N being replaced by L and T, are derived as follows:

- The variable numIs4N is set equal to ((availT && availL && predModeIntra == INTRA_LT_CCLM) ? 0 : 1).
- The variable startPosN is set equal to numSampN >> (2 + numIs4N).
- The variable pickStepN is set equal to Max(1, numSampN >> (1 + numIs4N)).
- If availN is equal to TRUE and predModeIntra is equal to INTRA_LT_CCLM or INTRA_N_CCLM, cntN is set equal to Min(numSampN, (1 + numIs4N) << 1), and pickPosN[pos] is set equal to (startPosN + pos * pickStepN), with pos = 0..(cntN – 1).
- Otherwise, cntN is set equal to 0.

The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:

- If both numSampL and numSampT are equal to 0, the following applies:

  predSamples[x][y] = 1 << ($BitDepth_C$ − 1) (8-162)

- Otherwise, the following ordered steps apply:
  1. The collocated luma samples pY[x][y] with x = 0..nTbW * 2 − 1, y= 0..nTbH * 2 − 1 are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x, yTbY + y).
  2. The neighbouring luma samples samples pY[x][y] are derived as follows:
     - When numSampL is greater than 0, the neighbouring left luma samples pY[x][y] with x = −1..−3, y = 0..2 * numSampL − 1, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY + x , yTbY +y).
     - When numSampT is greater than 0, the neighbouring top luma samples pY[x][y] with x = 0..2 * numSampT − 1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
     - When availTL is equal to TRUE, the neighbouring top-left luma samples pY[x][y] with x = −1, y = −1, −2, are set equal to the reconstructed luma samples prior to the deblocking filter process at the locations (xTbY+ x, yTbY + y).
  3. The down-sampled collocated luma samples pDsY[x][y] with x = 0..nTbW − 1, y = 0..nTbH − 1 are derived as follows:

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

  – pDsY[x][y] with x = 1..nTbW − 1, y = 1..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2*x][2*y-1] + pY[2*x-1][2*y] + 4*pY[2*x][2*y] + pY[2*x+1][2*y] + pY[2*x][2*y+1] + 4) >> 3 \quad (8\text{-}163)$$

  – If availL is equal to TRUE, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2*y-1] + pY[-1][2*y] + 4*pY[0][2*y] + pY[1][2*y] + pY[0][2*y+1] + 4) >> 3 \quad (8\text{-}164)$$

  – Otherwise, pDsY[0][y] with y = 1..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2*y-1] + 2*pY[0][2*y] + pY[0][2*y+1] + 2) >> 2 \quad (8\text{-}165)$$

  – If availT is equal to TRUE, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2*x][-1] + pY[2*x-1][0] + 4*pY[2*x][0] + pY[2*x+1][0] + pY[2*x][1] + 4) >> 3 \quad (8\text{-}166)$$

  – Otherwise, pDsY[x][0] with x = 1..nTbW − 1 is derived as follows:

$$pDsY[x][0] = (pY[2*x-1][0] + 2*pY[2*x][0] + pY[2*x+1][0] + 2) >> 2 \quad (8\text{-}167)$$

  – If availL is equal to TRUE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + pY[-1][0] + 4*pY[0][0] + pY[1][0] + pY[0][1] + 4) >> 3 \quad (8\text{-}168)$$

  – Otherwise if availL is equal to TRUE and availT is equal to FALSE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[-1][0] + 2*pY[0][0] + pY[1][0] + 2) >> 2 \quad (8\text{-}169)$$

  – Otherwise if availL is equal to FALSE and availT is equal to TRUE, pDsY[0][0] is derived as follows:

$$pDsY[0][0] = (pY[0][-1] + 2*pY[0][0] + pY[0][1] + 2) >> 2 \quad (8\text{-}170)$$

  – Otherwise (availL is equal to FALSE and availT is equal to FALSE), pDsY[0][0] is derived as follows:

$$pDsY[0][0] = pY[0][0] \quad (8\text{-}171)$$

– Otherwise, the following applies:

– pDsY[x][y] with x = 1..nTbW − 1, y = 0..nTbH − 1 is derived as follows:

$$pDsY[x][y] = (pY[2*x-1][2*y] + pY[2*x-1][2*y+1] + 2*pY[2*x][2*y] + 2*pY[2*x][2*y+1] + pY[2*x+1][2*y] + pY[2*x+1][2*y+1] + 4) >> 3 \quad (8\text{-}172)$$

– If availL is equal to TRUE, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[-1][2*y] + pY[-1][2*y+1] + 2*pY[0][2*y] + 2*pY[0][2*y+1] + pY[1][2*y] + pY[1][2*y+1] + 4) >> 3 \quad (8\text{-}173)$$

– Otherwise, pDsY[0][y] with y = 0..nTbH − 1 is derived as follows:

$$pDsY[0][y] = (pY[0][2*y] + pY[0][2*y+1] + 1) >> 1 \quad (8\text{-}174)$$

4. When numSampL is greater than 0, the selcted neighbouring left chroma samples pSelC[idx] are set equal to p[-1][pickPosL[idx]] with idx = 0..(cntL – 1), and the selected down-sampled neighbouring left luma samples pSelDsY[idx] with idx = 0..(cntL-1) are derived as follows:

– The variable y is set equal to pickPosL[idx].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If y > 0 || availTL == TRUE,

$$pSelDsY[idx] = (pY[-2][2*y-1] + pY[-3][2*y] + 4*pY[-2][2*y] + pY[-1][2*y] + pY[-2][2*y+1] + 4) >> 3 \quad (8\text{-}175)$$

– Otherwise,

$$pSelDsY[idx] = (pY[-3][0] + 2*pY[-2][0] + pY[-1][0] + 2) >> 2 \quad (8\text{-}177)$$

– Otherwise, the following applies:

$$pSelDsY[idx] = (pY[-1][2*y] + pY[-1][2*y+1] + 2*pY[-2][2*y] + 2*pY[-2][2*y+1] + pY[-3][2*y] + pY[-3][2*y+1] + 4) >> 3 \quad (8\text{-}178)$$

5. When numSampT is greater than 0, the selcted neighbouring top chroma samples pSelC[idx] are set equal to p[pickPosT[idx – cntL]][-1] with idx = cntL..(cntL + cntT – 1), and the down-sampled neighbouring top luma samples pSelDsY[idx] with idx = cntL..(cntL + cntT – 1) are specified as follows:

– The variable x is set equal to pickPosT[idx – cntL].

– If sps_cclm_colocated_chroma_flag is equal to 1, the following applies:

– If x > 0:

– If bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[2 * x][-3] + pY[2 * x - 1][-2] + 4 * pY[2 * x][-2] + pY[2 * x + 1][-2] + pY[2 * x][-1] + 4) >> 3 \quad (8\text{-}179)$$

– Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-1] + 2* pY[2 * x][-1] + pY[2 * x + 1][-1] + 2) >> 2 \quad (8\text{-}180)$$

– Otherwise:

– If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + pY[-1][-2] + 4 * pY[0][-2] + pY[1][-2] + pY[0][-1] + 4) >> 3 \quad (8\text{-}181)$$

– Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$pSelDsY[idx] = (pY[-1][-1] + 2* pY[0][-1] + pY[1][-1] + 2) >> 2 \quad (8\text{-}182)$$

– Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-3] + 2 * pY[0][-2] + pY[0][-1] + 2) >> 2 \quad (8\text{-}183)$$

– Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}184)$$

– Otherwise, the following applies:

– If x > 0:

– If bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-2] + pY[2 * x - 1][-1] + 2* pY[2 * x][-2] + 2*pY[2 * x][-1] + pY[2 * x + 1][-2] + pY[2 * x + 1][-1] + 4) >> 3 \quad (8\text{-}185)$$

– Otherwise (bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = (pY[2 * x - 1][-1] + 2* pY[2 * x][-1] + pY[2 * x + 1][-1] + 2) >> 2 \quad (8\text{-}186)$$

- Otherwise:
  - If availTL is equal to TRUE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[-1][-2] + pY[-1][-1] + 2* pY[0][-2] + 2*pY[0][-1] + pY[1][-2] + pY[1][-1] + 4) >> 3 \quad (8\text{-}187)$$

  - Otherwise if availTL is equal to TRUE and bCTUboundary is equal to TRUE, the following applies:

$$pSelDsY[idx] = (pY[-1][-1] + 2* pY[0][-1] + pY[1][-1] + 2) >> 2 \quad (8\text{-}188)$$

  - Otherwise if availTL is equal to FALSE and bCTUboundary is equal to FALSE, the following applies:

$$pSelDsY[idx] = (pY[0][-2] + pY[0][-1] + 1) >> 1 \quad (8\text{-}189)$$

  - Otherwise (availTL is equal to FALSE and bCTUboundary is equal to TRUE), the following applies:

$$pSelDsY[idx] = pY[0][-1] \quad (8\text{-}190)$$

6. When cntT+ cntL is not equal to 0, the variables minY, maxY, minC and maxC are derived as follows:
   -
   - When cntT+cntL is equal to 2, set pSelComp[3] equal to pSelComp [0], pSelComp[2] equal to pSelComp[1], pSelComp[0] equal to pSelComp [1], and pSelComp[1] equal to pSelComp[3], with Comp being replaced by DsY and C.
   - The arrays minGrpIdx[] and maxGrpIdx[] are set as: minGrpIdx[0] = 0, minGrpIdx[1] = 2, maxGrpIdx[0] = 1, maxGrpIdx[1] = 3.
   - If pSelDsY[minGrpIdx[0]] > pSelDsY[minGrpIdx[1]], Swap(minGrpIdx[0], minGrpIdx[1]).
   - If pSelDsY[maxGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(maxGrpIdx[0], maxGrpIdx[1]).
   - If pSelDsY[minGrpIdx[0]] > pSelDsY[maxGrpIdx[1]], Swap(minGrpIdx, maxGrpIdx).
   - If pSelDsY[minGrpIdx[1]] > pSelDsY[maxGrpIdx[0]], Swap(minGrpIdx[1], maxGrpIdx[0]).
   - maxY = (pSelDsY[maxGrpIdx[0]] + pSelDsY[maxGrpIdx[1]] + 1) >> 1.
   - maxC = (pSelC[maxGrpIdx[0]] + pSelC[maxGrpIdx[1]] + 1) >> 1.

– minY = (pSelDsY[minGrpIdx[0]] + pSelDsY[minGrpIdx[1]] + 1) >> 1.
– minC = (pSelC[minGrpIdx[0]] + pSelC[minGrpIdx[1]] + 1) >> 1.

7. The variables a, b, and k are derived as follows:
   – If numSampL is equal to 0, and numSampT is equal to 0, the following applies:

$$k = 0 \quad (8\text{-}208)$$

$$a = 0 \quad (8\text{-}209)$$

$$b = 1 << (BitDepth_C - 1) \quad (8\text{-}210)$$

   – Otherwise, the following applies:

$$diff = maxY - minY \quad (8\text{-}211)$$

     – If diff is not equal to 0, the following applies:

$$diffC = maxC - minC \quad (8\text{-}212)$$

$$x = Floor(Log2(diff)) \quad (8\text{-}213)$$

$$normDiff = ((diff << 4) >> x) \& 15 \quad (8\text{-}214)$$

$$x \mathrel{+}= (normDiff \ != \ 0) \ ? \ 1 \ : \ 0 \quad (8\text{-}215)$$

$$y = Floor(Log2(Abs (diffC))) + 1 \quad (8\text{-}216)$$

$$a = (diffC * (divSigTable[normDiff] | 8) + 2^{y-1}) >> y \quad (8\text{-}217)$$

$$k = ((3 + x - y) < 1) \ ? \ 1 \ : \ 3 + x - y \quad (8\text{-}218)$$

$$a = ((3 + x - y) < 1) \ ? \ Sign(a) * 15 \ : \ a \quad (8\text{-}219)$$

$$b = minC - ((a * minY) >> k) \quad (8\text{-}220)$$

where divSigTable[ ] is specified as follows:

$$divSigTable[ ] = \{ 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 \} \quad (8\text{-}221)$$

     – Otherwise (diff is equal to 0), the following applies:

$$k = 0 \quad (8\text{-}222)$$

$$a = 0 \quad (8\text{-}223)$$

$$b = minC \quad (8\text{-}224)$$

8. The prediction samples predSamples[x][y] with x = 0..nTbW − 1, y = 0.. nTbH − 1 are derived as follows:

$$predSamples[x][y] = Clip1C(((pDsY[x][y] * a) >> k) + b) \quad (8\text{-}225)$$

[00332] The examples described above may be incorporated in the context of the methods

described below, e.g., methods 1800 to 2930, which may be implemented at a video encoder and/or decoder.

[00333] FIG. 18 shows a flowchart of an exemplary method for video processing. The method 1800 includes, at step 1802, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on two chroma samples from a group of neighboring chroma samples, wherein the two chroma samples are selected from the group based on a position rule. The method 1800 further includes, at step 1804, performing the conversion based on the determining.

[00334] FIG. 19A shows a flowchart of an exemplary method for video processing. The method 1900 includes, at step 1902, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on selected chroma samples based on positions of the chroma samples, wherein the selected chroma samples are selected from a group of neighboring chroma samples. The method 1900 further includes, at step 1804, performing the conversion based on the determining.

[00335] FIG. 19B shows a flowchart of an exemplary method for video processing. The method 1910 includes, at step 1912, determining, for a current video block, a group of neighboring chroma samples used to derive a set of values for parameters of a linear model, wherein a width and a height of the current video block is W and H, respectively, and wherein the group of neighboring chroma samples comprises at least one sample that is located beyond 2×W above neighboring chroma samples or 2×H left neighboring chroma samples. The method 1910 further includes, at step 1914, performing, based on the linear model, a conversion between the current video block and a coded representation of a video including the current video block.

[00336] FIG. 20A shows a flowchart of an exemplary method for video processing. The method 2000 includes, at step 2002, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, multiple sets of parameters, wherein each set of parameters defines a cross-component linear model (CCLM) and is derived from a corresponding group of chroma samples at corresponding chroma sample positions. The method 2000 further includes, at step 2004, determining, based on the multiple sets of parameters, parameters for a final CCLM. The method 2000 further includes, at step

2006, performing the conversion based on the final CCLM.

[00337] FIG. 20B shows a flowchart of an exemplary method for video processing. The method 2010 includes, at step 2012, determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on maximum and minimum values of chroma and luma samples of N groups of chroma and luma samples selected from neighboring luma and chroma samples of the current video block. The method 2010 further includes, at step 2014, performing the conversion using the CCLM.

[00338] FIG. 21 shows a flowchart of an exemplary method for video processing. The method 2100 includes, at step 2102, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model that are completely determinable by two chroma samples and corresponding two luma samples. The method 2100 further includes, at step 2104, performing the conversion based on the determining.

[00339] FIG. 22 shows a flowchart of an exemplary method for video processing. The method 2200 includes, at step 2202, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model using a parameter table whose entries are retrieved according to two chroma sample values and two luma sample values. The method 220 further includes, at step 2204, performing the conversion based on the determining.

[00340] FIG. 23A shows a flowchart of an exemplary method for video processing. The method 2310 includes, at step 2312, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a final prediction P(x, y) of a chroma sample at a position (x, y) in the current video block as a combination of prediction results of multiple cross-component linear models (MCCLMs), wherein the MCCLMs are selected based on the position (x, y) of the chroma sample. The method 2310 further includes, at step 2314, performing the conversion based on the final prediction.

[00341] FIG. 23B shows a flowchart of an exemplary method for video processing. The method 2320 includes, at step 2322, performing, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a first determination regarding whether a first cross-component linear model (CCLM) that uses only left-neighboring samples is

used for predicting samples of the current video block and/or a second determination regarding whether a second cross-component linear model (CCLM) that uses only above-neighboring samples is used for predicting samples of the current video block. The method 2320 further includes, at step 2324, performing the conversion based on the first determination and/or the second determination.

[00342] FIG. 24A shows a flowchart of an exemplary method for video processing. The method 2410 includes, at step 2412, determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode. The method 2410 includes, at step 2414, performing the conversion based on the determining. In some implementations, wherein the flag is signaled to indicate whether the CCLM prediction mode is applied on the current video block, and wherein the CCLM prediction mode uses a linear mode to derive prediction values of a chroma component from another component.

[00343] FIG. 24B shows a flowchart of an exemplary method for video processing. The method 2420 includes, at step 2422, determining, for a conversion between a current video block of a video and a coded representation of the video, a coding order for one or more indications of a derived mode (DM mode) and a linear mode (LM mode) based on a coding mode of one or more neighboring blocks of the current video block. The method 2420 includes, at step 2424, performing the conversion based on the determining. In some implementations, a linear mode to derive prediction values of a chroma component from another component, and the DM mode derives intra prediction mode of a chroma component from another component.

[00344] FIG. 24C shows a flowchart of an exemplary method for video processing. The method 2430 includes, at step 2432, determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on refined neighboring luma samples and chroma samples of the current video block. The method 2430 includes, at step 2434, deriving prediction values of a chroma component of the current video block based on the parameters and refined internal luma samples of the current video block. The method 2430 includes, at step 2436, performing the conversion based on the prediction values.

[00345] FIG. 24D shows a flowchart of an exemplary method for video processing. The method 2440 includes, at step 2442, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on by selecting neighboring samples based on a position of a largest or a smallest neighboring sample. The method 2440 further includes, at step 2444, deriving prediction values of chroma samples of the current video block based on the parameters and internal luma samples of the current video block. The method 2440 further includes, at step 2446, performing the conversion based on the prediction values.

[00346] FIG. 24E shows a flowchart of an exemplary method for video processing. The method 2450 includes, at step 2452, determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on a main color component and a dependent color component, the main color component selected as one of a luma color component and a chroma color component and the dependent color component selected as the other of the luma color component and the chroma color component. The method 2450 further includes, at step 2454, performing the conversion based on the determining.

[00347] FIG. 25A shows a flowchart of an exemplary method for video processing. The method 2510 includes, at step 2512, performing downsampling on chroma and luma samples of a neighboring block of the current video block. The method 2510 further includes, at step 2514, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on the downsampled chroma and luma samples obtained from the downsampling. The method 2510 further includes, at step 2516, performing the conversion based on the determining.

[00348] FIG. 25B shows a flowchart of an exemplary method for video processing. The method 2520 includes, at step 2522, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or more chroma samples from a group of neighboring chroma samples, wherein the two or more chroma samples are selected based on a coding mode of the current video block. The method 2520 further includes, at step 2524, performing the conversion based on the determining.

[00349] FIG. 26A shows a flowchart of an exemplary method for video processing. The method

2610 includes, at step 2612, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on W available above-neighboring samples, W being an integer. The method 2610 further includes, at step 2614, performing the conversion based on the determining.

[00350] FIG. 26B shows a flowchart of an exemplary method for video processing. The method 2620 includes, at step 2622, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on H available left-neighboring samples of the current video block. The method 2620 further includes, at step 2622, performing the conversion based on the determining.

[00351] FIG. 27A shows a flowchart of an exemplary method for video processing. The method 2710 includes, at step 2712, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or four chroma samples and/or corresponding luma samples. The method 2710 further includes, at step 2714, performing the conversion based on the determining.

[00352] FIG. 27B shows a flowchart of an exemplary method for video processing. The method 2720 includes, at step 2722, selecting, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, chroma samples based on a position rule, the chroma samples used to derive parameters of a cross-component linear model (CCLM). The method 2720 further includes, at step 2724, performing the conversion based on the determining. In the example, the position rule specifies to select the chroma samples that are located within an above row and/or a left column of the current video block.

[00353] FIG. 28A shows a flowchart of an exemplary method for video processing. The method 2810 includes, at step 2812, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, positions at which luma samples are downsampled, wherein the downsampled luma samples are used to determine parameters of a cross-component linear model (CCLM) based on chroma samples and downsampled luma samples, wherein the downsampled luma samples are at positions corresponding to positions of the chroma samples that are used to derive the parameters of the CCLM. The method 2810 further includes,

at step 2814, performing the conversion based on the determining.

[00354] FIG. 28B shows a flowchart of an exemplary method for video processing. The method 2820 includes, at step 2822, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a method to derive parameters of a cross-component linear model (CCLM) using chroma samples and luma samples based on a coding condition associated with the current video block. The method 2820 further includes, at step 2824, performing the conversion based on the determining.

[00355] FIG. 28C shows a flowchart of an exemplary method for video processing. The method 2830 includes, at step 2832, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, whether to derive maximum values and/or minimum values of a luma component and a chroma component that are used to derive parameters of a cross-component linear model (CCLM) based on availability of a left-neighboring block and an above-neighboring block of the current video block. The method 2830 further includes, at step 2834, performing the conversion based on the determining.

[00356] FIG. 29A shows a flowchart of an exemplary method for video processing. The method 2910 includes, at step 2912, determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a coding tool using a linear model based on selected neighboring samples of the current video block and corresponding neighboring samples of a reference block. The method 2910 further includes, at step 2914, performing the conversion based on the determining.

[00357] FIG. 29B shows a flowchart of an exemplary method for video processing. The method 2920 includes, at step 2922, determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a local illumination compensation (LIC) tool based on N neighboring samples of the current video block and N corresponding neighboring samples of a reference block, wherein the N neighboring samples of the current video block are selected based on positions of the N neighboring samples. The method 2920 further includes, at step 2924, performing the conversion based on the determining. The LIC tool uses a linear model of illumination changes in the current video block during the conversion.

[00358] FIG. 29C shows a flowchart of an exemplary method for video processing. The method 2930 includes, at step 2932, determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component

linear model (CCLM) based on chroma samples and corresponding luma samples. The method 2930 further includes, at step 2934, performing the conversion based on the determining. In the example, some of the chroma samples are obtained by a padding operation and the chroma samples and the corresponding luma samples are grouped into two arrays G0 and G1, each array including two chroma samples and corresponding luma samples.

4 Example implementations of the disclosed technology

[00359] FIG. 30A is a block diagram of a video processing apparatus 3000. The apparatus 3000 may be used to implement one or more of the methods described herein. The apparatus 3000 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 3000 may include one or more processors 3002, one or more memories 3004 and video processing hardware 3006. The processor(s) 3002 may be configured to implement one or more methods (including, but not limited to, methods as shown FIGS. 18 to 29C) described in the present document. The memory (memories) 3004 may be used for storing data and code used for implementing the methods and techniques described herein. The video processing hardware 3006 may be used to implement, in hardware circuitry, some techniques described in the present document.

[00360] FIG. 30B is another example of a block diagram of a video processing system in which disclosed techniques may be implemented. FIG. 30B is a block diagram showing an example video processing system 3100 in which various techniques disclosed herein may be implemented. Various implementations may include some or all of the components of the system 3100. The system 3100 may include input 3102 for receiving video content. The video content may be received in a raw or uncompressed format, e.g., 8 or 10 bit multi-component pixel values, or may be in a compressed or encoded format. The input 3102 may represent a network interface, a peripheral bus interface, or a storage interface. Examples of network interface include wired interfaces such as Ethernet, passive optical network (PON), etc. and wireless interfaces such as Wi-Fi or cellular interfaces.

[00361] The system 3100 may include a coding component 3104 that may implement the various coding or encoding methods described in the present document. The coding component 3104 may reduce the average bitrate of video from the input 3102 to the output of the coding component 3104 to produce a coded representation of the video. The coding techniques are therefore sometimes called video compression or video transcoding techniques. The output of the

coding component 3104 may be either stored, or transmitted via a communication connected, as represented by the component 3106. The stored or communicated bitstream (or coded) representation of the video received at the input 3102 may be used by the component 3108 for generating pixel values or displayable video that is sent to a display interface 3110. The process of generating user-viewable video from the bitstream representation is sometimes called video decompression. Furthermore, while certain video processing operations are referred to as "coding" operations or tools, it will be appreciated that the coding tools or operations are used at an encoder and corresponding decoding tools or operations that reverse the results of the coding will be performed by a decoder.

[00362] Examples of a peripheral bus interface or a display interface may include universal serial bus (USB) or high definition multimedia interface (HDMI) or Displayport, and so on. Examples of storage interfaces include SATA (serial advanced technology attachment), PCI, IDE interface, and the like. The techniques described in the present document may be embodied in various electronic devices such as mobile phones, laptops, smartphones or other devices that are capable of performing digital data processing and/or video display.

[00363] In some embodiments, the video coding methods may be implemented using an apparatus that is implemented on a hardware platform as described with respect to FIG. 30A or 30B.

[00364] Various techniques and embodiments may be described using the following clause-based format.

[00365] The first set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 1.a-d and j.

[00366] 1. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on two chroma samples from a group of neighboring chroma samples, wherein the two chroma samples are selected from the group based on a position rule; and performing the conversion based on the determining.

[00367] 2. The method of clause 1, wherein the parameters have values that are determined based on luma samples of the two chroma samples.

[00368] 3. The method of clause 2, wherein the luma samples are downsampled used to derive the parameters of the cross-component linear model.

[00369] 4. The method of clause 1, wherein the parameters have values that are determined based on chroma samples at 2^S positions, S being an integer.

[00370] 5. The method of clause 1, wherein a top-left sample of the chroma block is (x, y), wherein a width and a height of the chroma block is W and H, respectively, and wherein the group of neighboring chroma samples comprises: sample A with coordinates (x − 1, y), sample B with coordinates (x − 1, y + H/2 − 1), sample C with coordinates (x − 1, y + H/2), sample D with coordinates (x − 1, y + H − 1), sample E with coordinates (x − 1, y + H), sample F with coordinates (x − 1, y + H + H/2 − 1), sample G with coordinates (x − 1, y + H + H/2), sample I with coordinates (x − 1, y + H + H − 1), sample J with coordinates (x, y − 1), sample K with coordinates (x + W/2 − 1, y − 1), sample L with coordinates (x + W/2, y − 1), sample M with coordinates (x + W − 1, y − 1), sample N with coordinates (x + W, y − 1), sample O with coordinates (x + W + W/2 − 1, y − 1), sample P with coordinates (x + W + W/2, y − 1), and sample Q with coordinates (x + W + W − 1, y − 1).

[00371] 6. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, D, J and M.

[00372] 7. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, B, C, D, J, K, L and M.

[00373] 8. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, I, J and Q.

[00374] 9. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, B, D, I, J, K, M and Q.

[00375] 10. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, B, D, F, J, K, M and O.

[00376] 11. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, C, G, I, J, L, P and Q.

[00377] 12. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, C, E, G, J, L, N and P.

[00378] 13. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples J, K, L and M.

[00379] 14. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples J, L, N and Q.

[00380] 15. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples J, K, L, M, N, O, P and Q.

[00381] 16. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples A, B, C, D, E, F, G, and I.

[00382] 17. The method of clause 5, wherein the position rule specifies that the two chroma samples are selected from the samples J, K, L, M, N, O, P, and Q.

[00383] 18. The method of clause 5, wherein the position rule specifies that one of the two chroma samples are selected from the samples A, B, C, D, E, F, G, and I and the other of the two chroma samples are selected from the samples J, K, L, M, N, O, P, and Q.

[00384] 19. The method of any of clauses 5-18, wherein the two chroma samples have identical corresponding luma values, the method further includes checking an additional chroma sample.

[00385] 20. The method of clause 1, wherein chroma samples in the group of neighboring chroma samples are searched to find the two chroma samples with minimum and maximum corresponding luma values to determine a first set of values for the parameters.

[00386] 21. The method of clause 1, wherein an above neighboring sample with coordinates (x, y) is in the group only if x%K=0, K being 2, 4, 6 or 8 and % being a modulo operator.

[00387] 22. The method of clause 1, wherein a left neighboring sample with coordinates (x, y) is in the group only if y%K=0, K being 2, 4, 6 or 8 and % being a modulo operator.

[00388] 23. The method of clause 1, wherein the two chroma samples are selected based on availabilities of neighboring blocks.

[00389] 24. The method of any of clauses 1-23, wherein the performing of the conversion includes generating the coded representation from the current block.

[00390] 25. The method of any of clauses 1-23, wherein the performing of the conversion includes generating the current block from the coded representation.

[00391] 26. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 25.

[00392] 27. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 25.

[00393] The second set of clauses describe certain features and aspects of the disclosed

techniques listed in the previous section, including, for example, Example 1.e-i and Example 9.

[00394] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on selected chroma samples based on positions of the chroma samples, wherein the selected chroma samples are selected from a group of neighboring chroma samples; and performing the conversion based on the determining.

[00395] 2. The method of clause 1, wherein at least one neighboring chroma sample does not belong to the selected chroma samples.

[00396] 3. The method of clause 1, wherein all of the selected chroma neighboring samples are left to the current block in a case that a prediction mode of the current video block is a first linear mode that uses left-neighboring samples only.

[00397] 4. The method of clause 1, wherein all of the selected chroma neighboring samples are above to the current block in a case that a prediction mode of the current video block is a second linear uses above-neighboring samples only.

[00398] 5. The method of any of clauses 1-4, wherein positions of the chroma samples are selected based on the width or the height of the chroma block.

[00399] 6. The method of clauses 1, wherein the positions of the chroma samples correspond to a signaling in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a slice header, a tile group header, a tile, a coding unit (CU), a coding tree unit (CTU) or a prediction unit (PU).

[00400] 7. The method of clause 1, wherein determining of the parameters is further based on a least mean squares method.

[00401] 8. The method of clause 1, wherein the determining of the parameters is further based on a two-point method.

[00402] 9. A method for video processing, comprising: determining, for a current video block, a group of neighboring chroma samples used to derive a set of values for parameters of a linear model, wherein a width and a height of the current video block is W and H, respectively, and wherein the group of neighboring chroma samples comprises at least one sample that is located beyond $2\times W$ above neighboring chroma samples or $2\times H$ left neighboring chroma samples; and performing, based on the linear model, a conversion between the current video block and a coded representation of a video including the current video block.

[00403] 10. The method of clause 8, wherein the current video block is coded using a linear intra prediction mode, wherein a top-left sample of the current video block is (x, y), wherein the at least one sample is $(x-1, y+d)$, wherein d is an integer in a range $[T, S]$, and wherein T and S are integers.

[00404] 11. The method of clause 9, wherein $T < 0$ and $S > (2\times H-1)$.

[00405] 12. The method of clause 9, wherein $T = -4$ and $S = 3\times H$.

[00406] 13. The method of clause 9, wherein $T = 0$ and $S = max(2\times W, W+H)$.

[00407] 14. The method of claim 9, wherein $T = 0$ and $S = 4\times H$.

[00408] 15. The method of clause 8, wherein the current video block is coded using a linear intra prediction mode, wherein a top-left sample of the current video block is (x, y), wherein the at least one sample is $(x+d, y-1)$, wherein d is an integer in a range $[T, S]$, and wherein T and S are integers.

[00409] 16. The method of clause 15, wherein $T < 0$ and $S > (2\times W-1)$.

[00410] 17. The method of clause 15, wherein $T = -4$ and $S = 3\times W$.

[00411] 18. The method of clause 15, wherein $T = 0$ and $S = max(2\times W, W+H)$.

[00412] 19. The method of clause 15, wherein $T = 0$ and $S = 4\times W$.

[00413] 20. The method of any of clauses 1-19, wherein the performing of the conversion includes generating the coded representation from the current block.

[00414] 21. The method of any of clauses 1-19, wherein the performing of the conversion includes generating the current block from the coded representation.

[00415] 22. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 21.

[00416] 23. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 21.

[00417] The third set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 2 and Example 5.

[00418] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, multiple sets of parameters, wherein each set of parameters defines a cross-component

linear model (CCLM) and is derived from a corresponding group of chroma samples at corresponding chroma sample positions; determining, based on the multiple sets of parameters, parameters for a final CCLM; and performing the conversion based on the final CCLM.

[00419] 2. The method of clause 1, wherein the parameters for the final CCLM are determined as an average of corresponding parameters in the multiple sets of parameters.

[00420] 3. The method of clause 1, wherein the multiple sets of parameters include a first set of ($\alpha1$, $\beta1$) and a second set of ($\alpha2$, $\beta2$) and a chroma prediction is calculated based on parameters of $\alpha1$, $\beta1$, $\alpha2$, $\beta2$.

[00421] 4. The method of clause 1, wherein the multiple sets of parameters are shifted and combined to form the final CCLM.

[00422] 5. The method of clause 1, wherein the multiple sets of parameters include a first set of ($\alpha1$, $\beta1$) that is derived from a first group of chroma samples and a second set of ($\alpha2$, $\beta2$) that is derived from a second group of chroma samples, the first group and the second group corresponding to different chroma sample positions from each other.

[00423] 6. The method of clause 5, wherein a top-left sample of the chroma block is (x, y) and a width and a height of the chroma block is W and H, respectively, and wherein the group of chroma samples comprises at least one of:

[00424] sample A with coordinates ($x - 1, y$),

[00425] sample B with coordinates ($x - 1, y + H/2 - 1$),

[00426] sample C with coordinates ($x - 1, y + H/2$),

[00427] sample D with coordinates ($x - 1, y + H - 1$),

[00428] sample E with coordinates ($x - 1, y + H$),

[00429] sample F with coordinates ($x - 1, y + H + H/2 - 1$),

[00430] sample G with coordinates ($x - 1, y + H + H/2$),

[00431] sample I with coordinates ($x - 1, y + H + H - 1$),

[00432] sample J with coordinates ($x, y - 1$),

[00433] sample K with coordinates ($x + W/2 - 1, y - 1$),

[00434] sample L with coordinates ($x + W/2, y - 1$),

[00435] sample M with coordinates ($x + W - 1, y - 1$),

[00436] sample N with coordinates ($x + W, y - 1$),

[00437] sample O with coordinates ($x + W + W/2 - 1, y - 1$),

[00438] sample P with coordinates (x + W + W/2, y − 1), or

[00439] sample Q with coordinates (x + W + W − 1, y − 1).

[00440] 7. The method of clause 6, wherein the first group includes Samples A and D and the second group includes Samples J and M or Samples E and I.

[00441] 8. The method of clause 6, wherein the first group includes Samples A and I and the second group includes Samples J and Q.

[00442] 9. The method of clause 6, wherein the first group includes Samples A and B and the second group includes Samples C and D.

[00443] 10. The method of clause 6, wherein the first group includes J and M and the second group includes N and Q.

[00444] 11. The method of clause 6, wherein the first group includes J and K and the second group includes L and M.

[00445] 12. A method of video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on maximum and minimum values of chroma and luma samples of N groups of chroma and luma samples selected from neighboring luma and chroma samples of the current video block; and performing the conversion using the CCLM.

[00446] 13. The method of clause 12, wherein the N groups of chroma and luma samples comprise $S_0, S_1, \ldots, S_m$ and wherein $1 \leq m \leq N-1$, wherein m and N are non-negative integers, wherein a maximum luma value is calculated as $maxL = f1(maxL_{S0}, maxL_{S1}, \ldots, maxL_{Sm})$, wherein $f1$ is a first function and $maxL_{Si}$ is a maximum luma value of a group S_i of the plurality of groups, wherein a maximum chroma value is calculated as $maxC = f2(maxC_{S0}, maxC_{S1}, \ldots, maxC_{Sm})$, wherein $f2$ is a second function and $maxC_{Si}$ is a chroma value of the group S_i corresponding to $maxL_{Si}$, wherein a minimum luma value is calculated as $minL = f3(minL_{S0}, minL_{S1}, \ldots, minL_{Sm})$, wherein $f3$ is a third function and $minL_{Si}$ is a minimum luma value of the group S_i, wherein a minimum chroma value is calculated as $minC = f4(minC_{S0}, minC_{S1}, \ldots, minC_{Sm})$, wherein $f4$ is a fourth function and $minC_{Si}$ is a chroma value of the group S_i, corresponding to $minL_{Si}$ and wherein the parameters of the linear model comprise α and β that are calculated as $\alpha = (maxC - minC) / (maxL - minL)$ and $\beta = minC - \alpha \times minL$.

[00447] 14. The method of clause 13, wherein $f1, f2, f3$ and $f4$ are averaging functions.

[00448] 15. The method of clause 13 or 14, wherein m = N−1.

[00449] 16. The method of clause 13 or 14, wherein m = 1, and wherein $S_1 = S_{N-1}$.

[00450] 17. The method of clause 13 or 14, wherein m = 0.

[00451] 18. The method of clause 13 or 14, wherein S_0 comprises samples from an above row of the current video block, and wherein S1 comprises samples from a left row of the current video block.

[00452] 19. The method of clause 13 or 14, wherein samples from an above row of the current video block have coordinates (x, y), wherein S_0 comprises a first portion of the samples, wherein each sample of the first portion satisfies x % P = Q, wherein S_1 comprises a second portion of the samples, wherein each sample of the second portion satisfies x % P ≠ Q, and wherein % is a modulo operator and P and Q are non-negative integers.

[00453] 20. The method of clause 13 or 14, wherein samples from a left row of the current video block have coordinates (x, y), wherein S_0 comprises a first portion of the samples, wherein each sample of the first portion satisfies y % P = Q, wherein S_1 comprises a second portion of the samples, wherein each sample of the second portion satisfies y % P ≠ Q, and wherein % is a modulo operator and P and Q are non-negative integers.

[00454] 21. The method of clause 19 or 20, wherein P = 2 and Q = 1, or P = 2 and Q = 0, or P = 4 and Q = 0.

[00455] 22. The method of any of clauses 12 to 14, wherein the chroma and luma samples comprise only a portion of the chroma and luma samples of the neighboring block.

[00456] 23. The method of any of clauses 13 to 22, wherein N is predetermined.

[00457] 24. The method of any of clauses 13 to 23, wherein N is signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, one or more largest coding units or one or more coding units.

[00458] 25. The method of clause 12, wherein the chroma and luma samples for each group are selected based on an availability of the neighboring block of the current video block.

[00459] 26. The method of clause 12, wherein the chroma and luma samples for each group are selected based on a width and a height of the current video block.

[00460] 27. The method of clause 12, wherein the chroma and luma samples for each group are selected based on values of the chroma and luma samples.

[00461] 28. The method of any of clauses 1-27, wherein the performing of the conversion includes generating the coded representation from the current block.

[00462] 29. The method of any of clauses 1-27, wherein the performing of the conversion includes generating the current block from the coded representation.

[00463] 30. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 29.

[00464] 31. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 29.

[00465] The fourth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 3.a-b and 3.d.

[00466] 1. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model that are completely determinable by two chroma samples and corresponding two luma samples; and performing the conversion based on the determining.

[00467] 2. The method of clause 1, wherein the two chroma samples are denoted as C0 and C1, the corresponding luma samples are denoted as L0 and L1, the parameters of the cross-component linear model are denoted α and β, and α and β are defined by equations: $\alpha = (C1 - C0) / (L1 - L0)$ and $\beta = C0 - \alpha \times L0$.

[00468] 3. The method of clause 2, wherein if L1 is equal to L0, then $\alpha = 0$.

[00469] 4. The method of clause 2, wherein if L1 is equal to L0, other intra prediction mode than the cross-component linear model mode is used.

[00470] 5. The method of clause 2, wherein α is determined by excluding a division operation.

[00471] 6. The method of clause 2, wherein α is determined using an operation without a look up table, the operation excluding a division operation.

[00472] 7. The method of clause 2, wherein the parameters of the cross-component linear model have values that are determined based on a value of (L1-L0).

[00473] 8. The method of clause 2, wherein $\alpha = \text{Shift}(C1 - C0, \text{Floor}(\log_2(L1 - L0)))$, wherein $\text{Shift}(x, s) = (x + \text{off}) >> s$ and off is an integer, wherein Floor(x) is a floor function that outputs an integer portion of x.

[00474] 9. The method of clause 2, wherein $\alpha = \text{Shift}(C1 - C0, \text{Ceiling}(\log_2(L1 - L0)))$,

wherein Shift(x, s) = (x + off) >> s and off is an integer, wherein Ceiling(x) is a ceiling function that outputs the least integer greater than or equal to x.

[00475] 10. The method of clause 8 or 9, wherein an operation for obtaining a value of log2(x) is performed by checking a position of a most significant digit of x.

[00476] 11. The method of clause 1, wherein the determining of the parameters is performed within K bits, K being 8, 10, 12, 16, 24 or 32.

[00477] 12. The method of clause 11, wherein an intermediate variable is clipped or right shifted to be within the K bits.

[00478] 13. The method of any of clauses 1-12, wherein the performing of the conversion includes generating the coded representation from the current block.

[00479] 14. The method of any of clauses 1-12, wherein the performing of the conversion includes generating the current block from the coded representation.

[00480] 15. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 14.

[00481] 16. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 14.

[00482] The fifth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 3.c.

[00483] 1. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model using a parameter table whose entries are retrieved according to two chroma sample values and two luma sample values; and performing the conversion based on the determining.

[00484] 2. The method of clause 1, wherein the parameter table has a size of V that is less than 2^P, P being an integer.

[00485] 3. The method of clause 1, wherein the parameter table has entries, each entry storing an F-bit integer number and F being 8 or 16.

[00486] 4. The method of clause 1, wherein the parameter table M[k] satisfies M[k-Z]=((1<<S)+Off)/k, S being an integer defining a precision, Off indicating an offset, Z being a

first value of the parameter table.

[00487] 5. The method of clause 1, wherein the two chroma samples are denoted as C0 and C1, the luma samples are denoted as L0 and L1, the parameters of the cross-component linear model are denoted α and β, and α and β are defined by equations: α = (C1 − C0) / (L1 − L0) and β = C0 − α×L0.

[00488] 6. The method of clause 5, wherein k = Shift(L1 − L0, W) and k is used to inquire an entry in the parameter table, and wherein Shift(x, s) = (x + off) >> s, off is an integer, and W is an width of the current video block.

[00489] 7. The method of clause 6, wherein α is zero when k-Z <0 or k-Z ≥ V, V indicating a size of the parameter table and Z indicating a first value of the parameter table.

[00490] 8. The method of clause 5, wherein $\alpha = Shift((C1 - C0) \times M[k - Z], D)$ or $\alpha = SignShift((C1 - C0) \times M[k - Z], D)$, and wherein Shift(x, s) = (x + off) >> s and SignShift (x, s) = (x + off) >> s, if x≥0, or -(-x + off) >> s, if x <0, off is an integer, and k indicating an index to inquire an entry in the parameter table.

[00491] 9. The method of clause 8, wherein k is derived based on a value of (L1-L0) and not based on a value of (C1-C0).

[00492] 10. The method of clause 8, wherein k is derived based on both values of (L1-L0) and (C1-C0).

[00493] 11. The method of clause 6, wherein k is valid within a range between kMin and kMax.

[00494] 12. The method of clause 8, wherein k = Shift(L1 − L0, W) and k is an index to inquire an entry in the parameter table, and wherein Shift(x, s) = (x + off) >> s, off is an integer, and W is an width of the current video block.

[00495] 13. The method of clause 8, wherein k is valid within a range between kMin and kMax, and k = L1-L0, if (L1-L0) ≤ kMax and k = Shift (L1-L0, W) if (L1=L0) > kMax, .

[00496] 14. The method of clause 8, wherein k is valid within a range between kMin and kMax, and k = Min (kMax, L1-L0) or k = Max (kMin, Min (kMax, L1-L0)).

[00497] 15. The method of clause 5, wherein (L1-L0) < 0 and the determining is performed to derive a value of '- α' instead of α.

[00498] 16. The method of clause 5, wherein (L1-L0) = 0 and α is set to a default value that is 0 or 1.

[00499] 17. The method of clause 5, wherein (L1-L0) = 2^E, E≥0, α = Shift ((C1-C0), E) or Singshift ((C1-C0), E), and wherein Shift(x, s) = (x + off) >> s and SignShift (x, s) = (x + off) >> s, if x≥0, or -(-x + off) >> s, if x <0.

[00500] 18. The method of any of clauses 1-17, wherein the performing of the conversion includes generating the coded representation from the current block.

[00501] 19. The method of any of clauses 1-17, wherein the performing of the conversion includes generating the current block from the coded representation.

[00502] 20. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 19.

[00503] 21. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 19.

[00504] The sixth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 4 and Example 6.

[00505] 1. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a final prediction P(x, y) of a chroma sample at a position (x, y) in the current video block as a combination of prediction results of multiple cross-component linear models (MCCLMs), wherein the MCCLMs are selected based on the position (x, y) of the chroma sample; and performing the conversion based on the final prediction.

[00506] 2. The method of clause 1, wherein the multiple cross-component linear models include a first linear model whose parameters are derived only from left-neighboring samples and a second linear model whose parameters are derived only from above-neighboring samples.

[00507] 3. The method of clause 1, wherein some of the chroma samples are predicted based on left-neighboring samples only and some of the chroma samples are predicted based on above-neighboring samples only.

[00508] 4. The method of clause 2 or 3, wherein the final prediction P (x, y) of the chroma sample is based on a weighted average of a prediction P1 (x, y) by the first linear model and a prediction P2(x, y) by the second linear model.

[00509] 5. The method of clause 4, wherein P (x, y) = w1×P1(x, y) + w2×P2(x, y), wherein

w1 and w2 are weights constrained by w1+w2=1.

[00510] 6. The method of clause 4, wherein P (x, y) = (w1*P1(x, y) + w2*P2(x, y)+Offset)>>shift, where offset is an integer including 0 or 1<<(shift-1), shift is an integer, and w1 and w2 are weights constrained by w1+w2=1<<shift.

[00511] 7. The method of clause 4, wherein P (x, y) = (w1*P1(x, y) + ((1<<shift)-w1) * P2(x, y) + Offset)>>shift, where offset is an integer including 0 or 1<<(shift-1), shift is an integer, and w1 and w2 are weights.

[00512] 8. The method of any of clauses 5-7, wherein values of w1 and w2 depend on the position (x, y).

[00513] 9. The method of any of clauses 5-7, wherein if x > y, then w1 < w2, wherein if x < y, then w1 > w2, and wherein if x = y, then w1 = w2.

[00514] 10. The method of any of clauses 5-7, wherein, for a case of x<y, a value of (w1-w2) increases if a value of (y-x) increases.

[00515] 11. The method of any of clauses 5-7, wherein, for a case of x>y, a value of (w2-w1) increases if a value of (x-y) increases.

[00516] 12. A method of video processing, comprising: performing, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a first determination regarding whether a first cross-component linear model (CCLM) that uses only left-neighboring samples is used for predicting samples of the current video block and/or a second determination regarding whether a second cross-component linear model (CCLM) that uses only above-neighboring samples is used for predicting samples of the current video block; and performing the conversion based on the first determination and/or the second determination.

[00517] 13. The method of clause 12, wherein the first CCLM is not applied for a case that satisfies W > K × H, K being a non-negative integer.

[00518] 14. The method of clause 12, wherein the second CCLM is not applied for a case that satisfies H> K × W, K being a non-negative integer.

[00519] 15. The method of clause 12, wherein a flag indicating the application of the first CCLM or the second CCLM is not signaled for a case that one of the first CCLM and the second CCLM is not applied.

[00520] 16. The method of any of clauses 1-15, wherein the performing of the conversion includes generating the coded representation from the current block.

[00521] 17. The method of any of clauses 1-15, wherein the performing of the conversion includes generating the current block from the coded representation.

[00522] 18. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 17.

[00523] 19. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 17.

[00524] The seventh set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Examples 7, 8, 11-13.

[00525] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode; and performing the conversion based on the determining, wherein the flag is signaled to indicate whether the CCLM prediction mode is applied on the current video block, and wherein the CCLM prediction mode uses a linear mode to derive prediction values of a chroma component from another component.

[00526] 2. The method of clause 1, wherein the context comprise a first context of the top-left neighboring block uses the CCLM prediction mode and comprises a second context different from the first context if the top-left neighboring block does not use the CCLM prediction mode.

[00527] 3. The method of clause 1, wherein the top-left neighboring block is unavailable, and wherein the CCLM prediction mode is considered to be enabled.

[00528] 4. The method of clause 1, wherein the top-left neighboring block is unavailable, and wherein the CCLM prediction mode is considered to be disabled.

[00529] 5. The method of clause 1, wherein the top-left neighboring block is intra-coded, and wherein the CCLM prediction mode is considered to be enabled.

[00530] 6. The method of clause 1, wherein the top-left neighboring block is intra-coded, and wherein the CCLM prediction mode is considered to be disabled.

[00531] 7. A method of video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, a coding order for one or

more indications of a derived mode (DM mode) and a linear mode (LM mode) based on a coding mode of one or more neighboring blocks of the current video block; and performing the conversion based on the determining, wherein the LM mode uses a linear mode to derive prediction values of a chroma component from another component, and the DM mode derives intra prediction mode of a chroma component from another component.

[00532] 8. The method of clause 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with the LM mode, and wherein an indication of the LM mode is coded first.

[00533] 9. The method of clause 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with the DM mode, and wherein an indication of the DM mode is coded first.

[00534] 10. The method of clause 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with a coding mode that is different from the LM mode, and wherein an indication of the DM mode is coded first.

[00535] 11. The method of any of clauses 7 to 10, wherein the one or more indications are signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, one or more largest coding units or one or more coding units.

[00536] 12. A method of video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on refined neighboring luma samples and chroma samples of the current video block; deriving prediction values of a chroma component of the current video block based on the parameters and refined internal luma samples of the current video block; and performing the conversion based on the prediction values.

[00537] 13. The method of clause 12, wherein neighboring luma samples and internal luma samples are downsampled before a refinement process.

[00538] 14. The method of clause 12, wherein a refinement process comprises a filtering process or a non-linear process.

[00539] 15. The method of clause 12, wherein the parameters for the linear model prediction are α and β, wherein $\alpha = (C1-C0) / (L1-L0)$ and $\beta = C0 - \alpha L0$, wherein C0 and C1 are derived from neighboring chroma samples, and wherein L0 and L1 are derived from neighboring luma

samples.

[00540] 16. The method of clause 15, wherein C0 and L0 are based on S neighboring chroma and luma samples, denoted {Cx1, Cx2, …, CxS} and {Lx1, Lx2, …, LxS}, respectively, wherein C1 and L1 are based on T neighboring chroma and luma samples, denoted {Cy1, Cy2, …, CyT} and {Ly1, Ly2, …, LyT}, respectively, wherein {Cx1, Cx2, …, CxS} are corresponding to {Lx1, Lx2, …, LxS}, wherein {Cy1, Cy2, …, CyT} are corresponding to {Ly1, Ly2, …, LyT}, wherein C0 = f0(Cx1, Cx2, …, CxS), L0 = f1(Lx1, Lx2, …, LxS), C1 = f2(Cy1, Cy2, …, CyT) and L1 = f1(Ly1, Ly2, …, LyT), and wherein f0, f1, f2 and f3 are functions.

[00541] 17. The method of clause 16, wherein f0 and f1 are a first function.

[00542] 18. The method of clause 16, wherein f2 and f3 are a second function.

[00543] 19. The method of clause 17, wherein f0, f1, f2 and f3 are a third function.

[00544] 20. The method of clause 19, wherein the third function is an averaging function.

[00545] 21. The method of clause 16, wherein S = T.

[00546] 22. The method of clause 16, wherein {Lx1, Lx2, …, LxS} are the smallest samples of a group of luma samples.

[00547] 23. The method of clause 16, wherein {Ly1, Ly2, …, LyT} are the largest samples of a group of luma samples.

[00548] 24. The method of clause 22 or 23, wherein the group of luma samples comprises all neighboring samples used in VTM-3.0 to derive the parameters of the linear model prediction.

[00549] 25. The method of clause 22 or 23, wherein the group of luma samples comprises a subset of neighboring samples used in VTM-3.0 to derive the parameters of the linear model prediction, and wherein the subset is not identical to all the neighboring samples.

[00550] 26. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on by selecting neighboring samples based on a position of a largest or a smallest neighboring sample; deriving prediction values of chroma samples of the current video block based on the parameters and internal luma samples of the current video block; and performing the conversion based on the prediction values.

[00551] 27. The method of clause 26, wherein the largest neighboring sample is located at a position (x0, y0), samples in a region (x0-d1, y0), (x0, y0-d2), (x0+d3, y0), (x0, y0+d4) are utilized

to select the neighboring samples, and wherein{d1, d2, d3, d4} depend on the position (x0, y0).

[00552] 28. The method of clause 26, wherein the smallest neighboring sample is located at a position (x1, y1), samples in a region (x1-d1, y1), (x1, y1-d2), (x1+d3, y1), (x1, y1+d4) are utilized to select the neighboring samples, and wherein{d1, d2, d3, d4} depend on the position (x1, y1).

[00553] 29. The method of any of clauses 26-28, wherein the neighboring samples represent a color component.

[00554] 30. A method of video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on a main color component and a dependent color component, the main color component selected as one of a luma color component and a chroma color component and the dependent color component selected as the other of the luma color component and the chroma color component; and performing the conversion based on the determining.

[00555] 31. The method of any of clauses 1-30, wherein the performing of the conversion includes generating the coded representation from the current block.

[00556] 32. The method of any of clauses 1-30, wherein the performing of the conversion includes generating the current block from the coded representation.

[00557] 33. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 32.

[00558] 34. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 32.

[00559] The eight set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Examples 10 and 14.

[00560] 1. A method for video processing, comprising: performing downsampling on chroma and luma samples of a neighboring block of the current video block; determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on the downsampled chroma and luma samples obtained from the downsampling; and performing the conversion based on the determining.

[00561] 2. The method of clause 1, wherein the current video block has a height (H) and a width (W) and the downsampling is based on the height or the width.

[00562] 3. The method of clause 1, wherein the downsampled chroma and luma samples are obtained before deriving the parameters of the CCLM including α and β, and wherein α and β are defined by equations: $\alpha = (C1 - C0) / (L1 - L0)$ and $\beta = C0 - \alpha \times L0$.

[00563] 4. The method of clause 1, wherein the number of left-neighboring samples used to derive the parameters of CCLM is same as the number of above-neighboring samples used to derive the parameters of CCLM.

[00564] 5. The method of clause 2, wherein $W < H$ or $W > H$.

[00565] 6. The method of clause 2, wherein whether to downsample the chroma and luma samples of a left-neighboring block or an above-neighboring block depends on a relative size of W and H.

[00566] 7. The method of clause 6, wherein the downsampling is performed on the chroma and luma samples of a left-neighboring block in a case of $H > W$.

[00567] 8. The method of clause 6, wherein the downsampling is performed on the chroma and luma samples of an above-neighboring block in a case of $W > H$.

[00568] 9. The method of clause 7, wherein a top-left sample of the current video block is R[0, 0], wherein the downsampled chroma samples comprise samples R[−1, K×H/W], and wherein K is a non-negative integer ranging from 0 to W−1.

[00569] 10. The method of clause 8, wherein a top-left sample of the current video block is R[0, 0], wherein the downsampled chroma samples comprise samples R[K×H/W, −1], and wherein K is a non-negative integer ranging from 0 to H−1.

[00570] 11. A method of video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or more chroma samples from a group of neighboring chroma samples, wherein the two or more chroma samples are selected based on a coding mode of the current video block; and performing the conversion based on the determining.

[00571] 12. The method of clause 11, wherein two or more luma samples corresponding to the two or more chroma samples are used to derive the parameters of the cross-component linear model.

[00572] 13. The method of clause 12, wherein the two or more luma samples are downsampled to derive the parameters of the cross-compoenent linear model.

[00573] 14. The method of clause 11, wherein the two or more chroma samples are selected based on availabilities of neighboring samples.

[00574] 15. The method of clause 11, wherein the two or more chroma samples are selected from one or more of a left column, an above row, an above-right row or a below-left column relative to the current video block.

[00575] 16. The method of clause 11, wherein the two or more chroma samples are selected based on a ratio of a height of the current video block to a width of the current video block.

[00576] 17. The method of clause 11, wherein the two or more chroma samples are selected based on whether a width or a height of the current video block is equal to K, K being an integer.

[00577] 18. The method of clause 17, wherein K = 2.

[00578] 19. The method of clause 11, wherein the coding mode of the current video block is a first linear mode that is different from a second linear mode that uses only left-neighboring samples and a third linear mode that uses only above-neighboring samples, wherein coordinates of a top-left sample of the current video block are (x, y), and wherein a width and a height of the current video block is W and H, respectively.

[00579] 20. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x, y-1), (x-1, y+H-1) and (x + W-1, y-1).

[00580] 21. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x, y-1), (x-1, y + H - H/W -1) and (x + W-1, y-1), and wherein H > W.

[00581] 22. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x, y-1), (x-1, y + H -1) and (x + W - W/H-1, y-1), and wherein H < W.

[00582] 23. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x, y-1), (x-1, y + H – max(1, H/W)) and (x + W- max(1, W/H), y-1).

[00583] 24. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x, y-1), (x + W/4, y-1), (x + 2*W/4, y-1) and (x + 3*W/4, y – 1).

[00584] 25. The method of clause 19, wherein the two or more chroma samples comprise

samples with coordinates (x, y-1), (x + W/4, y-1), (x + 3*W/4, y – 1) and (x + W-1, y -1).

[00585] 26. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x, y-1), (x + (2W)/4, y-1), (x + 2*(2W)/4, y-1) and (x + 3*(2W)/4, y – 1).

[00586] 27. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x, y-1), (x + (2W)/4, y-1), (x + 3*(2W)/4, y – 1) and (x + (2W)-1, y-1).

[00587] 28. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x -1, y + H/4), (x -1, y + 2*H/4) and (x -1, y + 3*H/4).

[00588] 29. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x - 1, y+ 2*H/4), (x -1, y + 3*H/4) and (x -1, y + H-1).

[00589] 30. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x -1, y + (2H)/4), (x -1, y + 2*(2H)/4) and (x -1, y + 3*(2H)/4).

[00590] 31. The method of clause 19, wherein the two or more chroma samples comprise samples with coordinates (x-1, y), (x - 1, y+ 2*(2H)/4), (x -1, y + 3*(2H)/4) and (x -1, y + (2H)-1).

[00591] 32. The method of any of clauses 20 to 31, wherein exactly two samples are selected to determine the parameters of the CCLM.

[00592] 33. The method of any of clauses 1-32, wherein the performing of the conversion includes generating the coded representation from the current block.

[00593] 34. The method of any of clauses 1-32, wherein the performing of the conversion includes generating the current block from the coded representation.

[00594] 35. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 34.

[00595] 36. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 34.

[00596] The ninth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Examples 16 and 17.

[00597] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the

video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on W available above-neighboring samples, W being an integer; and performing the conversion based on the determining.

[00598] 2. The method of clause 1, wherein W is set to i) a width of the current video block, ii) L times the width of the current video block, L being an integer, iii) a sum of a height of the current video block and a width of the current video block, or iv) a sum of the width of the current video block and the number of available top-right neighboring samples.

[00599] 3. The method of clause 1, wherein W depends on an availability of at least one of an above-neighboring block or a left-neighboring block of the current video block.

[00600] 4. The method of clause 1, wherein W depends on a coding mode of the current video block.

[00601] 5. The method of clause 2, wherein L has a value depending on an availability of a top-right block or a top left sample that is located adjacent to the current video block.

[00602] 6. The method of clause 1, wherein the chroma samples are selected based on a first position offset value (F) and a step value (S) that depend on W.

[00603] 7. The method of clause 6, wherein a top left sample has a coordinate (x0, y0) and the selected chroma samples have coordinates (x0+F+K×S, y0-1), K being an integer between 0 and kMax.

[00604] 8. The method of clause 6, wherein F = W/P or F = W/P + offset, P being an integer.

[00605] 9. The method of clause 8, wherein F= W >> (2 + numIs4T), wherein numIs4T is equal to 1 in a case that there are four neighboring samples selected within an above neighboring row and otherwise numIs4T is equal to 0.

[00606] 10. The method of clause 6, wherein S = W/Q, Q being an integer.

[00607] 11. The method of clause 6, wherein S is not less than 1.

[00608] 12. The method of clause 10 or 11, wherein S= Max(1, W >> (1 + numIs4T)), wherein numIs4T is equal to 1 in a case that there are four neighboring samples selected within an above neighboring row and otherwise numIs4T is equal to 0.

[00609] 13. The method of clause 9 or 12, wherein numIs4T is equal to 1 in a case that above neighboring samples are available, left neighboring samples are available, and the current video block is coded with a normal CCLM that is different from a first CCLM using only left-neighboring samples, and different from a second CCLM using only above-neighboring samples.

[00610] 14. The method of clause 6, wherein F = S/R, R being an integer.

[00611] 15. The method of clause 6, wherein S = F/Z, Z being an integer.

[00612] 16. The method of any of clauses 7-15, wherein at least one of Kmax, F, S, or offset depends on a prediction mode of the current video block that is one of a first CCLM using only left-neighboring samples, a second CCLM using only above-neighboring samples, a third CCLM using both left-neighboring and above-neighboring samples, or other modes that are different from the first CCLM, the second CCLM, and the third CCLM.

[00613] 17. The method of any of clauses 7-15, wherein at least one of Kmax, F, S, or offset depends on a width and/or a height of the current video block.

[00614] 18. The method of any of clauses 7-15, wherein at least one of Kmax, F, S, or offset depends on availabilities of neighboring samples.

[00615] 19. The method of any of clauses 7-15, wherein at least one of Kmax, F, S, or offset depends on W.

[00616] 20. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on H available left-neighboring samples of the current video block; and performing the conversion based on the determining.

[00617] 21. The method of clause 20, wherein H is set to one of i) a height of the current video block, ii) L times the height of the current video block, L being an integer, iii) a sum of a height of the current video block and a width of the current video block, or iv) a sum of the height of the current video block and the number of available left-bottom neighboring samples..

[00618] 22. The method of claim 20, wherein H depends on an availability of at least one of an above-neighboring block or a left-neighboring block of the current video block.

[00619] 23. The method of clause 20, wherein H depends on a coding mode of the current video block.

[00620] 24. The method of clause 21, wherein L has a value depending on an availability of a below-left block or a below-left sample that is located adjacent to the current video block.

[00621] 25. The method of clause 20, wherein the chroma samples are selected based on a first position offset value (F) and a step value (S) that depend on H.

[00622] 26. The method of clause 25, wherein a top-left sample has a coordinate (x0, y0) and

the selected chroma samples have coordinates (x0-1, y0+F+K×S), K being an integer between 0 and kMax.

[00623] 27. The method of clause 25, wherein F = H/P or F = H/P + offset, P being an integer.

[00624] 28. The method of clause 27, wherein F= H >> (2 + numIs4L), wherein numIs4L is equal to 1 in a case that there is four neighboring samples selected within the left neighboring column; otherwise it is equal to 0.

[00625] 29. The method of clause 25, wherein S = H/Q, Q being an integer.

[00626] 30. The method of clause 25, wherein S is not less than 1.

[00627] 31. The method of clause 29 or 30, wherein S= Max(1, H >> (1 + numIs4L)), wherein numIs4L is equal to 1 in a case that there are four neighboring samples selected within a left neighboring column and otherwise numIs4L is equal to 0.

[00628] 32. The method of clause 28 or 31, wherein numIs4L is equal to 1 in a case that above neighboring samples are available, left neighboring samples are available, and the current video block is coded with a normal CCLM that is different from a first CCLM using only left-neighboring samples, and different from a second CCLM using only above-neighboring samples.

[00629] 33. The method of clause 25, wherein F = S/R, R being an integer.

[00630] 34. The method of clause 25, wherein S = F/Z, Z being an integer.

[00631] 35. The method of any of clauses 26-34, wherein at least one of Kmax, F, S, or offset depends on a prediction mode of the current video block that is one of a first CCLM using only left-neighboring samples, a second CCLM using only above-neighboring samples, a third CCLM using both left-neighboring and above-neighboring samples, or other modes that are different from the first CCLM, the second CCLM, and the third CCLM.

[00632] 36. The method of any of clauses 26-34, wherein at least one of Kmax, F, S, or offset depends on a width and/or a height of the current video block.

[00633] 37. The method of any of clauses 26-34, wherein at least one of Kmax, F, S, or offset depends on H.

[00634] 38. The method of any of clauses 26-34, wherein at least one of Kmax, F, S, or offset depends on availabilities of neighboring samples.

[00635] 39. The method of clause 20, wherein H is set to a sum of the height of the current video block and a width of the current video block in a case that an above-right neighboring block of the current video block is available.

[00636] 40. The method of clause 20, wherein in a case that left neighboring samples are unavailable, the selected chroma samples have the height H regardless of whether the current video block has a first CCLM using only above-neighboring samples or not.

[00637] 41. The method of clause 1, wherein W is set to a sum of the height of the current video block and a width of the current video block in a case that a below-left neighboring block of the current video block is available.

[00638] 42. The method of clause 1, wherein in a case that above neighboring samples are unavailable, the selected chroma samples have the number of W regardless of whether the current video block has a first CCLM using only left-neighboring samples or not.

[00639] 43. The method of any of clauses 1-42, wherein the performing of the conversion includes generating the coded representation from the current block.

[00640] 44. The method of any of clauses 1-42, wherein the performing of the conversion includes generating the current block from the coded representation.

[00641] 45. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 44.

[00642] 46. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 44.

[00643] The tenth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Examples 18 and 19.

[00644] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or four chroma samples and/or corresponding luma samples; and performing the conversion based on the determining.

[00645] 2. The method of clause 1, wherein the corresponding luma samples are obtained by down-sampling.

[00646] 3. The method of clause 1, wherein the parameters of the CCLM includes maxY/maxC and minY/minC.

[00647] 4. The method of clause 3, wherein the two chroma samples are selected to derive maxY/maxC and minY/minC, and wherein minY is set to be the smaller luma sample value, minC

is its corresponding chroma sample value, maxY is set to be the larger luma sample value, and maxC is its corresponding chroma sample value.

[00648] 5. The method of clause 3, wherein the four chroma samples are selected to derive maxY/maxC and minY/minC, and wherein the four chroma samples and the corresponding luma samples are divided into two arrays G0 and G1, each array including two chroma samples and their corresponding luma samples.

[00649] 6. The method of clause 5, wherein the two arrays G0 and G1 include one of following sets:

[00650] i) G0={S0, S1}, G1={S2, S3},

[00651] ii) G0={S1, S0}, G1={S3, S2},

[00652] iii) G0={S0, S2}, G1={S1, S3},

[00653] iv) G0={S2, S0}, G1={S3, S1},

[00654] v) G0={S1, S2}, G1={S0, S3},

[00655] vi) G0={S2, S1}, G1={S3, S0},

[00656] vii) G0={S0, S3}, G1={S1, S2},

[00657] viii) G0={S3, S0}, G1={S2, S1},

[00658] ix) G0={S1, S3}, G1={S0, S2},

[00659] x) G0={S3, S1}, G1={S2, S0},

[00660] xi) G0={S3, S2}, G1={S0, S1}, or

[00661] xii) G0={S2, S3}, G1={S1, S0}, and

[00662] wherein S0, S1, S2, S3 include the four chroma samples, respectively, and further includes corresponding luma samples, respectively.

[00663] 7. The method of clause 6, wherein upon a comparison of two luma sample values of G0[0] and G0[1], a chroma sample and its corresponding luma sample of G0[0] are swamped with those of G0[1].

[00664] 8. The method of clause 7, wherein the chroma sample and its corresponding luma sample of G0[0] are swamped with those of G0[1] in a case that a luma sample value of G0[0] is greater than a luma sample value of G0[1].

[00665] 9. The method of clause 6, wherein upon a comparison of two luma sample values of G1[0] and G1[1], a chroma sample and its corresponding luma sample of G1[0] are swamped with those of G1[1].

[00666] 10. The method of clause 9, wherein the chroma sample and its corresponding luma sample of G1[0] are swamped with those of G1[1] in a case that a luma sample value of G1[0] is greater than a luma sample value of G1[1].

[00667] 11. The method of clause 6, wherein upon a comparison of two luma sample values of G0[0] and G1[1], chroma samples and its corresponding luma samples of G0[0] or G0[1] are swamped with those of G1[0] or G1[1].

[00668] 12. The method of clause 11, wherein the chroma samples and its corresponding luma samples of G0[0] or G0[1] are swamped with those of G1[0] or G1[1] in a case that a luma sample value of G0[0] is greater than a luma sample value of G1[1].

[00669] 13. The method of clause 6, wherein upon a comparison of two luma sample values of G0[1] and G1[0], a chroma sample and its corresponding luma sample of G0[1] are swamped with those of G1[0].

[00670] 14. The method of clause 13, wherein the chroma sample and its corresponding luma sample of G0[1] are swamped with those of G1[0] in a case that a luma sample value of G0[1] is greater than a luma sample value of G1[0].

[00671] 15. The method of clause 6, wherein upon a comparison of two luma sample values of G0[0], G0[1], G1[0], and G1[1], following swamping operations are conducted in an order: i) a swamping operation of a chroma sample and its corresponding luma sample of G0[0] with those of G0[1], ii) a swamping operation of a chroma sample and its corresponding luma sample of G1[0] with those of G1[1], iii) a swamping operation of chroma samples and its corresponding luma samples of G0[0] or G0[1] with those of G1[0] or G1[1], and iv) a swamping operation of a chroma sample and its corresponding luma sample of G0[1] with those of G1[0].

[00672] 16. The method of clause 6, wherein maxY is calculated as an average of luma sample values of G0[0] and G0[1] or an average of luma sample values of G1[0] and G1[1], and maxC is calculated as an average of chroma sample values of G0[0] and G0[1] or an average of chroma sample values of G1[0] and G1[1].

[00673] 17. The method of clause 6, wherein minY is calculated as an average of luma sample values of G0[0] and G0[1] or an average of luma sample values of G1[0] and G1[1], and minC is calculated as an average of chroma sample values of G0[0] and G0[1] or an average of chroma sample values of G1[0] and G1[1].

[00674] 18. The method of clause 16 or 17, wherein the calculations of maxY and maxC or the

calculations of minY and minC are conducted after any one of swamping operations that are performed upon a comparison of two luma sample values of G0[0], G0[1], G1[0] and G1[1], wherein the swamping operations includes: i) a swamping operation of chroma sample and its corresponding luma sample of G1[0] with those of G1[1], ii) a swamping operation of chroma samples and its corresponding luma samples of G0[0] or G0[1] with those of G1[0] or G1[1], and iii) a swamping operation of a chroma sample and its corresponding luma sample of G0[1] with those of G1[0].

[00675] 19. The method of clause 1, wherein in a case that there are only two chroma samples are available, a padding is performed on the two available chroma samples to provide the four chroma samples.

[00676] 20. The method of clause 19, wherein the four chroma samples include the two available chroma samples and two padding chroma samples that are copied from the two available chroma samples.

[00677] 21. The method of clause 6, wherein S0, S1, S2, S3 are chroma samples and corresponding luma samples are selected in a given order within an above row and/or a left column of the current video block.

[00678] 22. A method for video processing, comprising: selecting, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, chroma samples based on a position rule, the chroma samples used to derive parameters of a cross-component linear model (CCLM); and performing the conversion based on the determining, wherein the position rule specifies to select the chroma samples that are located within an above row and/or a left column of the current video block.

[00679] 23. The method of clause 22, wherein the above row and the left column have W samples and H samples, respectively, W and H being a width and a height of the current video block, respectively.

[00680] 24. The method of clause 22, wherein the position rule is applied for the current video block coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00681] 25. The method of clause 22, wherein the position rule specifies to select the chroma samples that are located within the above row and an above-right row of the current video block,

and wherein the above row and the above-right row have W samples and H samples, respectively, W and H being a width and a height of the current video block, respectively.

[00682] 26. The method of clause 25, wherein only available samples within the above row and the above-right row are selected.

[00683] 27. The method of clause 25, wherein the position rule is applied for the current video block coded with a first CCLM mode that uses only above-neighboring samples to derive the CCLM.

[00684] 28. The method of clause 25, wherein the position rule is applied to a case that the above-row is available and the left column is unavailable and that the current video block is coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00685] 29. The method of any of clauses 23-28, wherein numSampT is set based on a rule specifying that numSampT is set equal to nTbW in a case that above neighboring samples are available and numSampT is set equal to 0 in a case that the above neighboring samples are not available, and wherein numSampT represents the number of chroma samples within an above neighboring row used to derive the parameters of a cross-component linear model and nTbW represents an width of the current video block.

[00686] 30. The method of clause 29, wherein the rule is applied for the current video block coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00687] 31. The method of any of clauses 23-28, wherein numSampT is set based on a rule specifying that numSampT is set equal to nTbW + Min(numTopRight, nTbH) in a case that above neighboring samples are available and the current video block is coded with a first CCLM mode that uses only above-neighboring samples to derive the CCLM, and that otherwise the numSampT is set equal to 0, and wherein numSampT represents the number of chroma samples within an above neighboring row used to derive the parameters of the cross-component linear model, nTbW and nTbH represent a width and a height of the current block, respectively, and numTopRight represents the number of available top right neighgoring samples.

[00688] 32. The method of clause 31, wherein the rule is applied for the current video block not

coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00689] 33. The method of clause 22, wherein the position rule specifies to select the chroma samples that are located within the left column and a below-left column of the current video block, and wherein the left column and the below-left column have H samples and W samples, respectively, W and H being a width and a height of the current video block, respectively.

[00690] 34. The method of clause 33, wherein only available samples within the left column and the below-left column are selected.

[00691] 35. The method of claim 33, wherein the position rule is applied for the current video block coded with a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00692] 36. The method of clause 33, wherein the position rule is applied to a case that the above-row is unavailable and the left-column is available and that the current video block is coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00693] 37. The method of any of clauses 33-36, wherein numSampL is set based on a rule specifying that numSampL is set equal to nTbH in a case that left neighboring samples are available and otherwise numSampL is set equal to 0, and wherein numSampL represents the number of chroma samples within a left neighbouring column used to derive parameters of the cross-component linear model and nTbH represents a height of the current video block.

[00694] 38. The method of clause 37, wherein the rule is applied for the current video block coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00695] 39. The method of any of clauses 33-36, wherein numSampL is set based on a rule specifying that numSampL is set equal to nTbH + Min(numLeftBelow, nTbW) in a case that left neighbouring samples are available and the current video block is coded with a second CCLM mode that uses only left-neighboring samples to derive the CCLM and that otherwise numSampL is set equal to 0, and wherein numSampL represents the number of chroma samples within a left

neighboring column used to derive the parameters of the cross-component linear model, nTbW and nTbH represent a width and a height of the current block, respectively, and numLeftBelow represents the number of available below-left neighgoring samples.

[00696] 40. The method of clause 39, wherein the rule is applied for the current video block not coded with a normal CCLM mode that is different from a first CCLM mode that uses only above-neighboring samples to derive the CCLM and from a second CCLM mode that uses only left-neighboring samples to derive the CCLM.

[00697] 41. The method of any of clauses 22-40, wherein luma samples corresponding to selected chroma samples are used to derive the parameters of the cross-component linear model.

[00698] 42. The method of clause 41, wherein the luma samples are derived by downsampling.

[00699] 43. The method of any of clauses 1-42, wherein the performing of the conversion includes generating the coded representation from the current block.

[00700] 44. The method of any of clauses 1-42, wherein the performing of the conversion includes generating the current block from the coded representation.

[00701] 45. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 44.

[00702] 46. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 44.

[00703] The eleventh set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Examples 20, 21, 22.

[00704] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, positions at which luma samples are downsampled, wherein the downsampled luma samples are used to determine parameters of a cross-component linear model (CCLM) based on chroma samples and downsampled luma samples, wherein the downsampled luma samples are at positions corresponding to positions of the chroma samples that are used to derive the parameters of the CCLM; and performing the conversion based on the determining.

[00705] 2. A method of clause 1, wherein luma samples are not downsampled at a position which is out of the current video block and is not used to determine the parameters of the CCLM.

[00706] 3. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a method to derive parameters of a cross-component linear model (CCLM) using chroma samples and luma samples based on a coding condition associated with the current video block; and performing the conversion based on the determining.

[00707] 4. The method of clause 3, wherein the coding condition corresponds to a color format of the current video block.

[00708] 5. The method of clause 4, wherein the color format is 4:2:0 or 4:4:4.

[00709] 6. The method of clause 3, wherein coding condition corresponds to a color representation method of the current video block.

[00710] 7. The method of clause 6, wherein the color representation method is a RGB or YCbCr.

[00711] 8. The method of clause 3, wherein the chroma samples are downsampled and the determining depends on locations of downsampled chroma samples.

[00712] 9. The method of clause 3, wherein the method to derive parameters comprises determining the parameters of the CCLM based on the chroma samples and the luma samples that are selected from a group of neighboring chroma samples based on a position rule.

[00713] 10. The method of clause 3, wherein the method to derive parameters comprises determining the parameters of the CCLM based on maximum and minimum values of the chroma samples and the luma samples.

[00714] 11. The method of clause 3, wherein the method to derive parameters comprises determining the parameters of the CCLM that are completely determinable by two chroma samples and corresponding two luma samples.

[00715] 12. The method of clause 3, wherein the method to derive parameters comprises determining the parameters of the CCLM using a parameter table whose entries are retrieved according to two chroma sample values and two luma sample values.

[00716] 13. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, whether to derive maximum values and/or minimum values of a luma component and a chroma component that are used to derive parameters of a cross-component linear model (CCLM) based on availability of a left-neighboring block and an above-neighboring block of the current video

block; and performing the conversion based on the determining.

[00717] 14. The method of clause 13, wherein the maximum values and/or the minimum values are not derived in a case that the left-neighboring block and the above-neighboring block are unavailable.

[00718] 15. The method of clause 13, wherein the determining determines based on a number of available neighboring samples of the current video block, and wherein the available neighboring samples are used to derive the parameters of the cross-component linear model.

[00719] 16. The method of clause 15, wherein the maximum values and/or the minimum values are not derived in a case of numSampL == 0 and numSampT == 0, the numSampL and the numSampT indicating a number of available neighboring samples from the left-neighboring block and a number of available neighboring samples from the above-neighboring block, respectively, and wherein the available neighboring samples from the left-neighboring block and the available neighboring samples from the above-neighboring bock are used to derive the parameters of the cross-component linear model.

[00720] 17. The method of clause 15, wherein the maximum values and/or the minimum values are not derived in a case of numSampL + numSampT == 0, the numSampL and the numSampT indicating a number of available neighboring samples from the left-neighboring block and a number of available neighboring samples from the above-neighboring block, respectively, and wherein the available neighboring samples from the left-neighboring block and the available neighboring samples from the above-neighboring bock are used to derive the parameters of the cross-component linear model.

[00721] 18. The method of any of clauses 1-17, wherein the performing of the conversion includes generating the coded representation from the current block.

[00722] 19. The method of any of clauses 1-17, wherein the performing of the conversion includes generating the current block from the coded representation.

[00723] 20. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 19.

[00724] 21. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 19.

[00725] The twelfth set of clauses describe certain features and aspects of the disclosed techniques listed in the previous section, including, for example, Example 23.

[00726] 1. A method for video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a coding tool using a linear model based on selected neighboring samples of the current video block and corresponding neighboring samples of a reference block; and performing the conversion based on the determining.

[00727] 2. The method of clause 1, wherein the coding tool is a local illumination compensation (LIC) tool that includes using a linear model of illumination changes in the current video block during the conversion.

[00728] 3. The method of clause 2, wherein the neighboring samples of the current video block and the neighboring samples of the reference block are selected based on a position rule.

[00729] 4. The method of clause 2, wherein the parameters of the coding tool are determined based on maximum and minimum values of the neighboring samples of the current video block and the neighboring samples of the reference block.

[00730] 5. The method of clause 2, wherein the parameters of the coding tool are determined using a parameter table whose entries are retrieved according to two neighboring samples of the current video block and two neighboring samples of the reference block.

[00731] 6. The method of clause 2, wherein the neighboring samples of the current video block and the neighboring samples of the reference block are downsampled to derive the parameters of the coding tool.

[00732] 7. The method of clause 2, wherein the neighboring samples used to derive parameters of the LIC tool excludes samples at certain positions in an above row and/or a left column of the current video block.

[00733] 8. The method of clause 2, wherein a top-left sample of the current video block has coordinates with (x0, y0), a sample with coordinates (x0, y0-1) is not used to derive parameters of the LIC tool.

[00734] 9. The method of clause 2, wherein a top-left sample of the current video block has coordinates with (x0, y0), a sample with coordinates (x0-1, y0) is not used to derive parameters of the LIC tool.

[00735] 10. The method of clause 7, wherein the certain positions depend on an availability of

the above row and/or the left column.

[00736] 11. The method of clause 7, wherein the certain positions depend on a block dimension of the current video block.

[00737] 12. The method of clause 1, wherein the determining depends on an availability of the above row and/or the left column.

[00738] 13. The method of clause 2, wherein N neighboring samples of the current video block and N neighboring samples of the reference block are used to derive the parameters of the LIC tool.

[00739] 14. The method of clause 13, wherein N is 4.

[00740] 15. The method of clause 13, wherein N neighboring samples of the current video block includes N/2 samples from an above row of the current video block and N/2 samples from a left column of the current video block.

[00741] 16. The method of clause 13, wherein N is equal to min (L, T), T being a total number of available neighboring samples of the current video block and L being an integer.

[00742] 17. The method of clause 13, wherein the N neighboring samples are selected based on a same rule that is applicable to select samples to derive parameters of the CCLM.

[00743] 18. The method of clause 13, wherein N neighboring samples are selected based on a same rule that is applicable to select samples to derive parameters of a first mode of the CCLM that uses above-neighboring samples only.

[00744] 19. The method of clause 13, wherein N neighboring samples are selected based on a same rule that is applicable to select samples to derive parameters of a second mode of the CCLM that uses left-neighboring samples only.

[00745] 20. The method of clause 13, wherein the N neighboring samples of the current video block are selected based on availabilities of an above row or a left column of the current video block.

[00746] 21. A method for video processing, comprising: determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a local illumination compensation (LIC) tool based on N neighboring samples of the current video block and N corresponding neighboring samples of a reference block, wherein the N neighboring samples of the current video block are selected based on positions of the N neighboring samples; and performing the conversion based on the determining, wherein the LIC tool uses a linear model

of illumination changes in the current video block during the conversion.

[00747] 22. The method of clause 21, wherein the N neighboring samples of the current video block are selected based on a width and a height of the current video block.

[00748] 23. The method of clause 21, wherein the N neighboring samples of the current video block are selected based on availabilities of neighboring blocks of the current video block.

[00749] 24. The method of claim 21, wherein the N neighboring samples of the current video block are selected with a first position offset value (F) and a step value (S) that depend on a dimension of the current video block and availabilities of neighboring blocks.

[00750] 25. The method of any of clauses 1-24, wherein the current video block is affine-coded.

[00751] 26. A method for video processing, comprising: determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on chroma samples and corresponding luma samples; and performing the conversion based on the determining, wherein some of the chroma samples are obtained by a padding operation and the chroma samples and the corresponding luma samples are grouped into two arrays G0 and G1, each array including two chroma samples and corresponding luma samples.

[00752] 27. The method of clause 26, wherein in a case that a sum of cntT and cntL is equal to 2, following operations are performed in an order: i) pSelComp[3] is set equal to pSelComp[0], ii) pSelComp[2] is set equal to pSelComp[1], iii) pSelComp[0] is set equal to pSelComp[1], and iv) pSelComp[1] is set equal to pSelComp[3], wherein cntT and cntL indicate the number of samples selected from an above neighboring block and a left neighboring block, respectively, and wherein pSelComp[0] to pSelComp[3] indicate pixel values of color components of corresponding samples that are selected.

[00753] 28. The method of clause 26, wherein the determining of the parameters includes initializing values of G0[0], G0[1], G1[0], and G1[1].

[00754] 29. The method of clause 28, wherein G0[0] = 0, G0[1] = 2, G1[0] = 1, and G1[1] = 3.

[00755] 30. The method of clause 28, wherein the determining of the parameters further includes, after the initializing of the values, upon a comparison of two luma sample values of G0[0] and G0[1], swamping a chroma sample and its corresponding luma sample of G0[0] with those of G0[1].

[00756] 31. The method of clause 30, wherein a chroma sample and its corresponding luma

sample of G0[0] are swamped with those of G0[1] in a case that a luma sample value of G0[0] is greater than a luma sample value of G0[1].

[00757] 32. The method of clause 28, wherein the determining of the parameters further includes, after the initializing of the values, upon a comparison of two luma sample values of G1[0] and G1[1], swamping a chroma sample and its corresponding luma sample of G1[0] with those of G1[1].

[00758] 33. The method of clause 32, wherein a chroma sample and its corresponding luma sample of G1[0] are swamped with those of G1[1] in a case that a luma sample value of G1[0] is greater than a luma sample value of G1[1].

[00759] 34. The method of clause 28, wherein the determining of the parameters further includes, after the initializing of the values, upon a comparison of two luma sample values of G0[0] and G1[1], swamping chroma samples and its corresponding luma samples of G0[0] or G0[1] with those of G1[0] or G1[1].

[00760] 35. The method of clause 34, wherein chroma samples and its corresponding luma samples of G0[0] or G0[1] are swamped with those of G1[0] or G1[1] in a case that a luma sample value of G0[0] is greater than a luma sample value of G1[1].

[00761] 36. The method of clause 28, wherein the determining of the parameters further includes, after the initializing of the values, upon a comparison of two luma sample values of G0[1] and G1[0], swamping a chroma sample and its corresponding luma sample of G0[1] with those of G1[0].

[00762] 37. The method of clause 36, wherein a chroma sample and its corresponding luma sample of G0[1] are swamped with those of G1[0] in a case that a luma sample value of G0[1] is greater than a luma sample value of G1[0].

[00763] 38. The method of clause 28, wherein the determining of the parameters further includes, after the initializing of the values, upon a comparison of two luma sample values of G0[0], G0[1], G1[0], and G1[1], performing following swamping operations in an order: i) a swamping operation of chroma sample and its corresponding luma sample of G0[0] with those of G0[1], ii) a swamping operation of chroma sample and its corresponding luma sample of G1[0] with those of G1[1], iii) a swamping operation of chroma samples and its corresponding luma samples of G0[0] or G0[1] with those of G1[0] or G1[1], and iv) a swamping operation of a chroma sample and its corresponding luma sample of G0[1] with those of G1[0].

[00764] 39. The method of any of clauses 1-38, wherein the performing of the conversion includes generating the coded representation from the current block.

[00765] 40. The method of any of clauses 1-38, wherein the performing of the conversion includes generating the current block from the coded representation.

[00766] 41. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of clauses 1 to 40.

[00767] 42. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of clauses 1 to 40.

[00768] From the foregoing, it will be appreciated that specific embodiments of the presently disclosed technology have been described herein for purposes of illustration, but that various modifications may be made without deviating from the scope of the invention. Accordingly, the presently disclosed technology is not limited except as by the appended claims.

[00769] Implementations of the subject matter and the functional operations described in this patent document can be implemented in various systems, digital electronic circuitry, or in computer software, firmware, or hardware, including the structures disclosed in this specification and their structural equivalents, or in combinations of one or more of them. Implementations of the subject matter described in this specification can be implemented as one or more computer program products, i.e., one or more modules of computer program instructions encoded on a tangible and non-transitory computer readable medium for execution by, or to control the operation of, data processing apparatus. The computer readable medium can be a machine-readable storage device, a machine-readable storage substrate, a memory device, a composition of matter effecting a machine-readable propagated signal, or a combination of one or more of them. The term “data processing unit” or “data processing apparatus” encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

[00770] A computer program (also known as a program, software, software application,

script, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[00771] The processes and logic flows described in this specification can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

[00772] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Computer readable media suitable for storing computer program instructions and data include all forms of nonvolatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[00773] It is intended that the specification, together with the drawings, be considered exemplary only, where exemplary means an example. As used herein, the use of “or” is intended

to include “and/or”, unless the context clearly indicates otherwise.

[00774] While this patent document contains many specifics, these should not be construed as limitations on the scope of any invention or of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular inventions. Certain features that are described in this patent document in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

[00775] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. Moreover, the separation of various system components in the embodiments described in this patent document should not be understood as requiring such separation in all embodiments.

[00776] Only a few implementations and examples are described and other implementations, enhancements and variations can be made based on what is described and illustrated in this patent document.

CLAIMS

What is claimed is:

1. A method for video processing, comprising:

determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode; and

performing the conversion based on the determining, wherein the flag is signaled to indicate whether the CCLM prediction mode is applied on the current video block, and

wherein the CCLM prediction mode uses a linear mode to derive prediction values of a chroma component from another component.

2. The method of claim 1, wherein the context comprise a first context of the top-left neighboring block uses the CCLM prediction mode and comprises a second context different from the first context if the top-left neighboring block does not use the CCLM prediction mode.

3. The method of claim 1, wherein the top-left neighboring block is unavailable, and wherein the CCLM prediction mode is considered to be enabled.

4. The method of claim 1, wherein the top-left neighboring block is unavailable, and wherein the CCLM prediction mode is considered to be disabled.

5. The method of claim 1, wherein the top-left neighboring block is intra-coded, and wherein the CCLM prediction mode is considered to be enabled.

6. The method of claim 1, wherein the top-left neighboring block is intra-coded, and wherein the CCLM prediction mode is considered to be disabled.

7. A method of video processing, comprising:

determining, for a conversion between a current video block of a video and a coded representation of the video, a coding order for one or more indications of a derived mode (DM mode) and a linear mode (LM mode) based on a coding mode of one or more neighboring blocks of the current video block; and

performing the conversion based on the determining;

wherein the LM mode uses a linear mode to derive prediction values of a chroma component from another component, and the DM mode derives intra prediction mode of a chroma component from another component.

8. The method of claim 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with the LM mode, and wherein an indication of the LM mode is coded first.

9. The method of claim 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with the DM mode, and wherein an indication of the DM mode is coded first.

10. The method of claim 7, wherein a top-left neighboring block of the one or more neighboring blocks is coded with a coding mode that is different from the LM mode, and wherein an indication of the DM mode is coded first.

11. The method of any of claims 7 to 10, wherein the one or more indications are signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, one or more largest coding units or one or more coding units.

12. A method of video processing:

determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component

prediction based on refined neighboring luma samples and chroma samples of the current video block;

deriving prediction values of a chroma component of the current video block based on the parameters and refined internal luma samples of the current video block; and

performing the conversion based on the prediction values.

13. The method of claim 12, wherein neighboring luma samples and internal luma samples of the current video block are down-sampled before a refinement process.

14. The method of claim 12, wherein a refinement process comprises a filtering process or a non-linear process.

15. The method of claim 12, wherein the parameters for the linear model prediction are α and β, wherein $\alpha = (C1-C0) / (L1-L0)$ and $\beta = C0 - \alpha L0$, wherein C0 and C1 are derived from neighboring chroma samples, and wherein L0 and L1 are derived from neighboring luma samples.

16. The method of claim 15, wherein C0 and L0 are based on S neighboring chroma and luma samples, denoted {Cx1, Cx2, …, CxS} and {Lx1, Lx2, …, LxS}, respectively, wherein C1 and L1 are based on T neighboring chroma and luma samples, denoted {Cy1, Cy2, …, CyT} and {Ly1, Ly2, …, LyT}, respectively,

wherein {Cx1, Cx2, …, CxS} are corresponding to {Lx1, Lx2, …, LxS},

wherein {Cy1, Cy2, …, CyT} are corresponding to {Ly1, Ly2, …, LyT},

wherein C0 = f0(Cx1, Cx2, …, CxS), L0 = f1(Lx1, Lx2, …, LxS), C1 = f2(Cy1, Cy2, …, CyT) and L1 = f1(Ly1, Ly2, …, LyT), and

wherein f0, f1, f2 and f3 are functions.

17. The method of claim 16, wherein f0 and f1 are a first function.

18. The method of claim 16, wherein f2 and f3 are a second function.

19. The method of claim 17, wherein f0, f1, f2 and f3 are a third function.

20. The method of claim 19, wherein the third function is an averaging function.

21. The method of claim 16, wherein S = T.

22. The method of claim 16, wherein {Lx1, Lx2, …, LxS} are the smallest samples of a group of luma samples.

23. The method of claim 16, wherein {Ly1, Ly2, …, LyT} are the largest samples of a group of luma samples.

24. The method of claim 22 or 23, wherein the group of luma samples comprises all neighboring samples used in VTM-3.0 to derive the parameters of the linear model prediction.

25. The method of claim 22 or 23, wherein the group of luma samples comprises a subset of neighboring samples used in VTM-3.0 to derive the parameters of the linear model prediction, and wherein the subset is not identical to all the neighboring samples.

26. A method of video processing;

determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on by selecting neighboring samples based on a position of a largest or a smallest neighboring sample;

deriving prediction values of chroma samples of the current video block based on the parameters and internal luma samples of the current video block; and

performing the conversion based on the prediction values.

27. The method of claim 26, wherein the largest neighboring sample is located at a position (x0, y0), samples in a region (x0-d1, y0), (x0, y0-d2), (x0+d3, y0), (x0, y0+d4) are utilized to select the neighboring samples, and wherein{d1, d2, d3, d4} depend on the position (x0, y0).

28. The method of claim 26, wherein the smallest neighboring sample is located at a position (x1, y1), samples in a region (x1-d1, y1), (x1, y1-d2), (x1+d3, y1), (x1, y1+d4) are utilized to select the neighboring samples, and wherein{d1, d2, d3, d4} depend on the position (x1, y1).

29. The method of any of claims 26-28, wherein the neighboring samples represent a color component.

30. A method of video processing,

determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on a main color component and a dependent color component, the main color component selected as one of a luma color component and a chroma color component and the dependent color component selected as the other of the luma color component and the chroma color component; and

performing the conversion based on the determining.

31. The method of any of claims 1-30, wherein the performing of the conversion includes generating the coded representation from the current block.

32. The method of any of claims 1-30, wherein the performing of the conversion includes generating the current block from the coded representation.

33. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method recited in one or more of claims 1 to 32.

34. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method recited in one or more of claims 1 to 32.

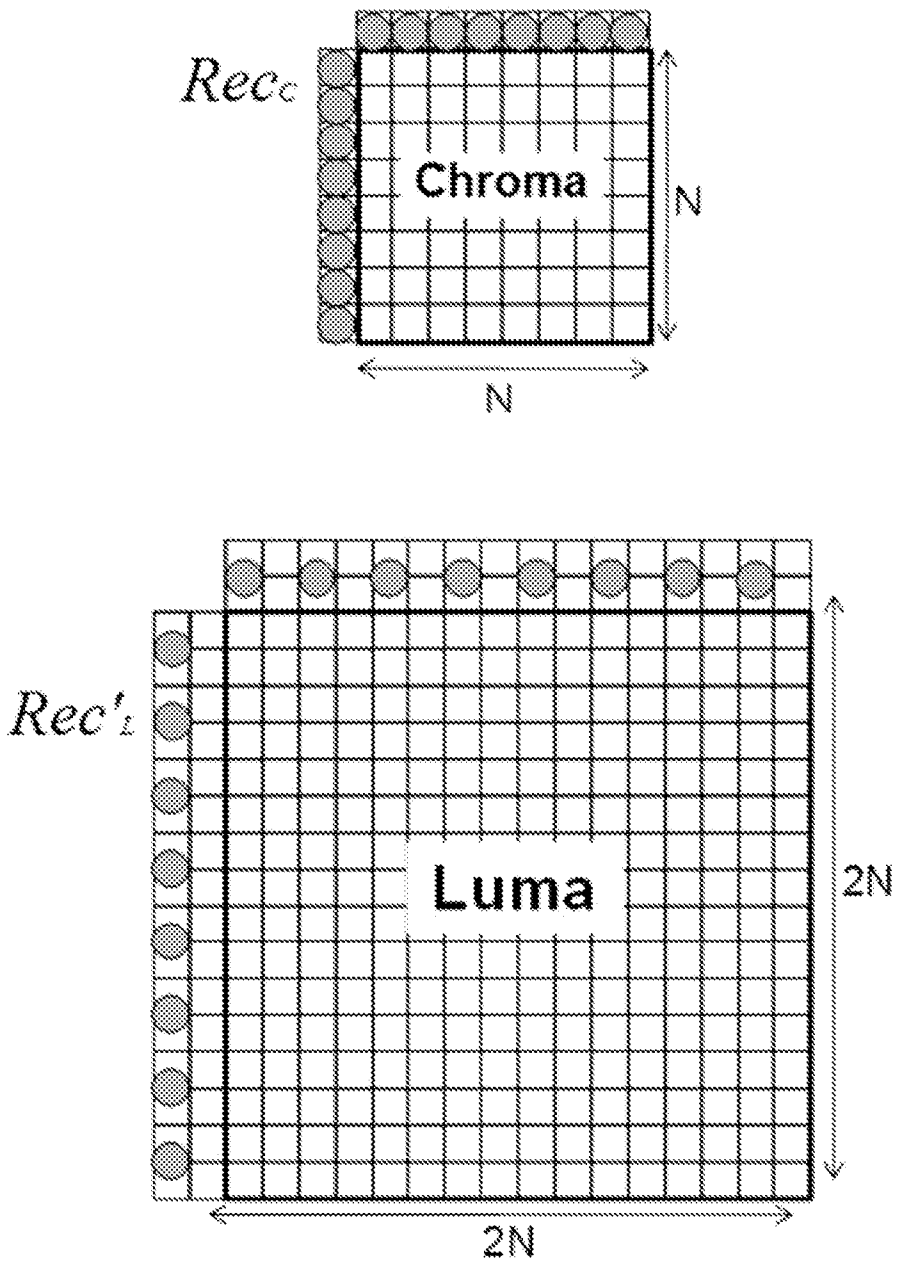

FIG. 1

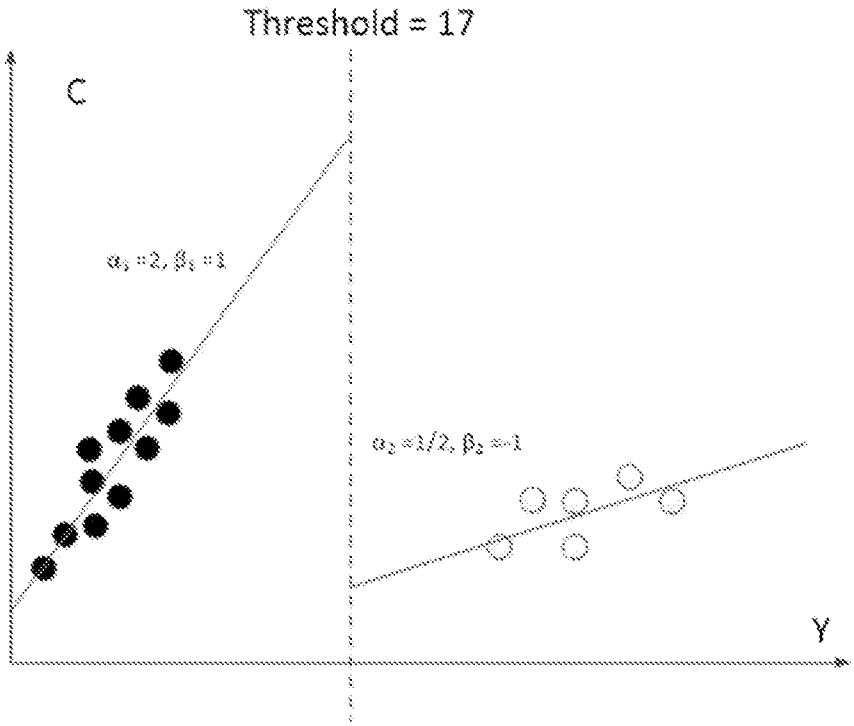

FIG. 2

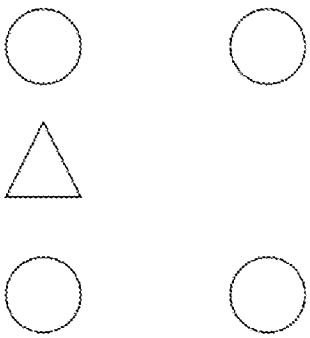

FIG. 3A

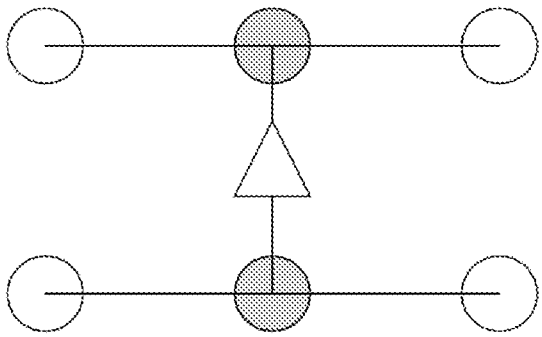

FIG. 3B

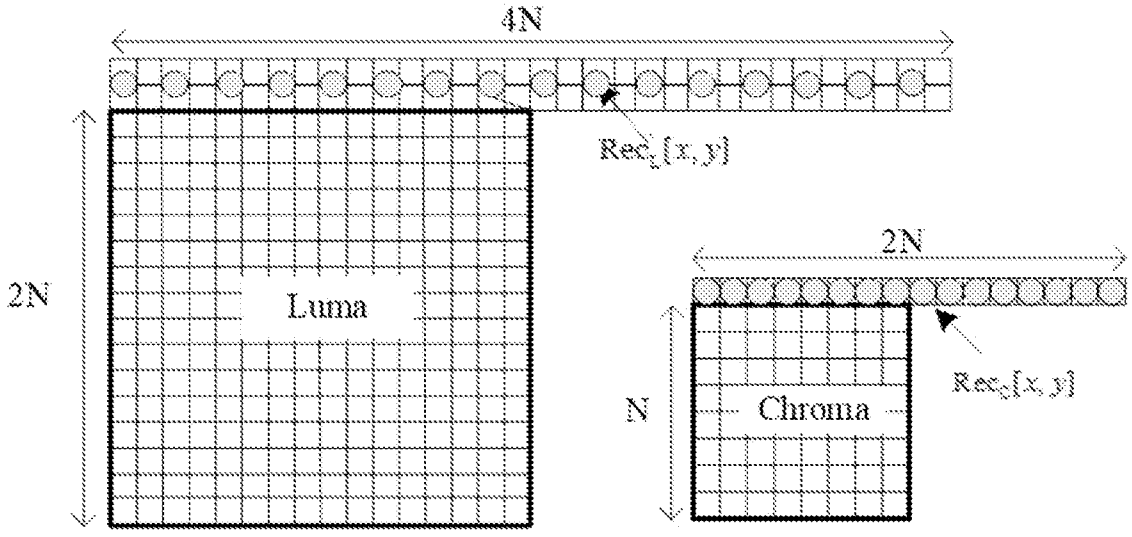
4N
2N
Luma
$Rec_L[x, y]$
2N
N
Chroma
$Rec_C[x, y]$

FIG. 4A

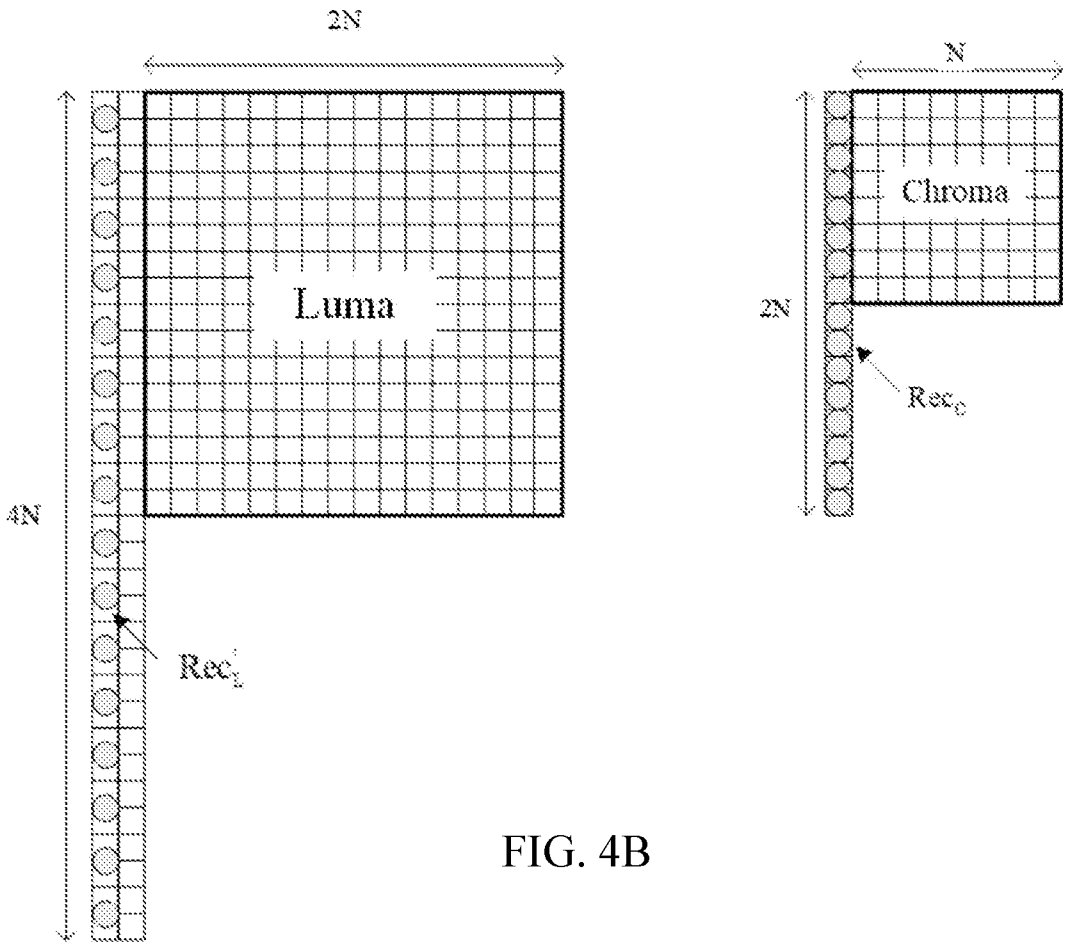
2N
4N
Luma
Rec_L'
N
2N
Chroma
Rec_C

FIG. 4B

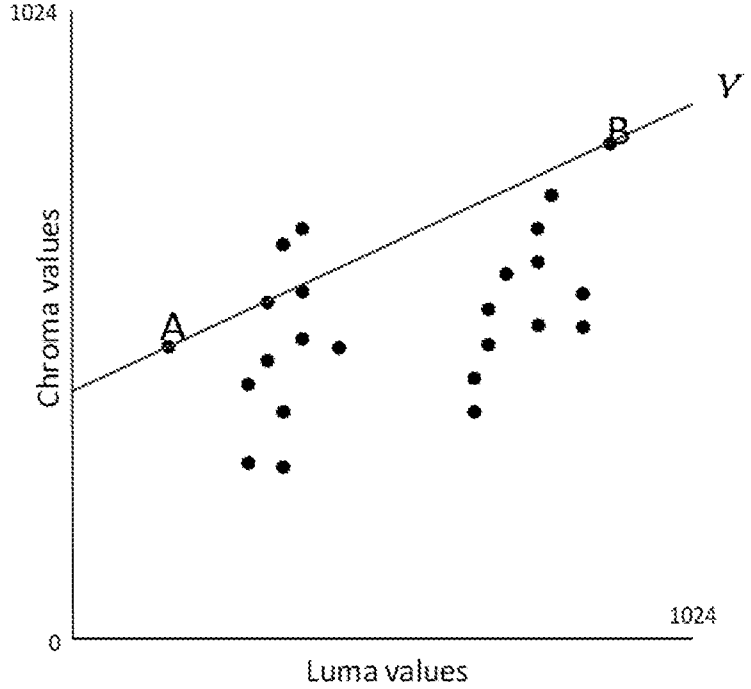
1024
Y
B
A
Chroma values
0
1024
Luma values

FIG. 5

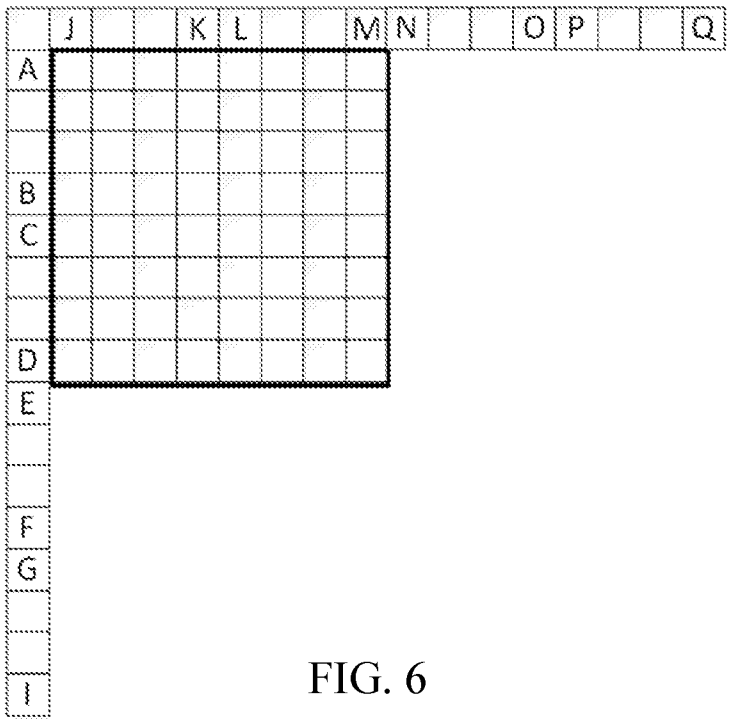
J
K
L
M
N
O
P
Q
A
B
C
D
E
F
G
I

FIG. 6

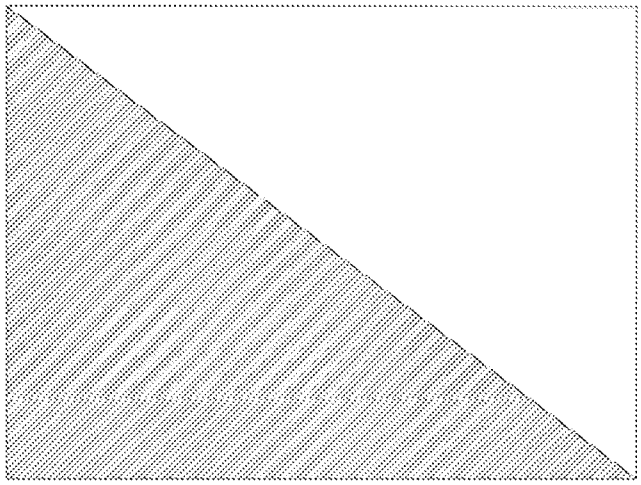

FIG. 7

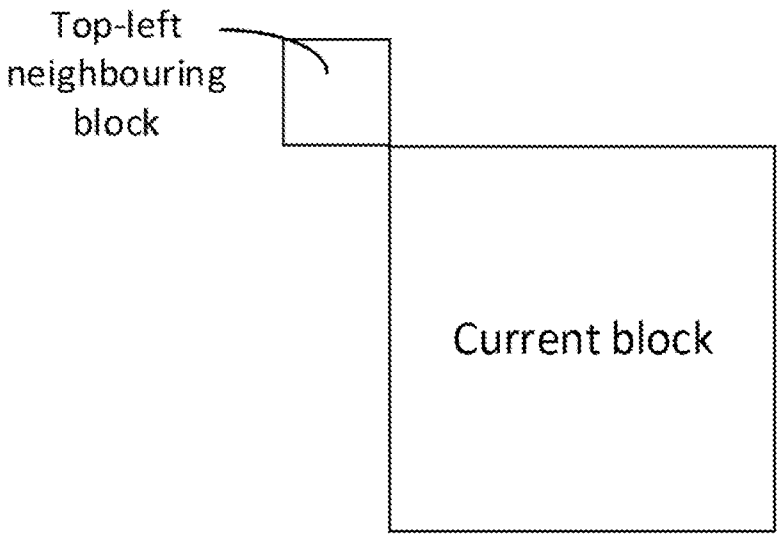

FIG. 8

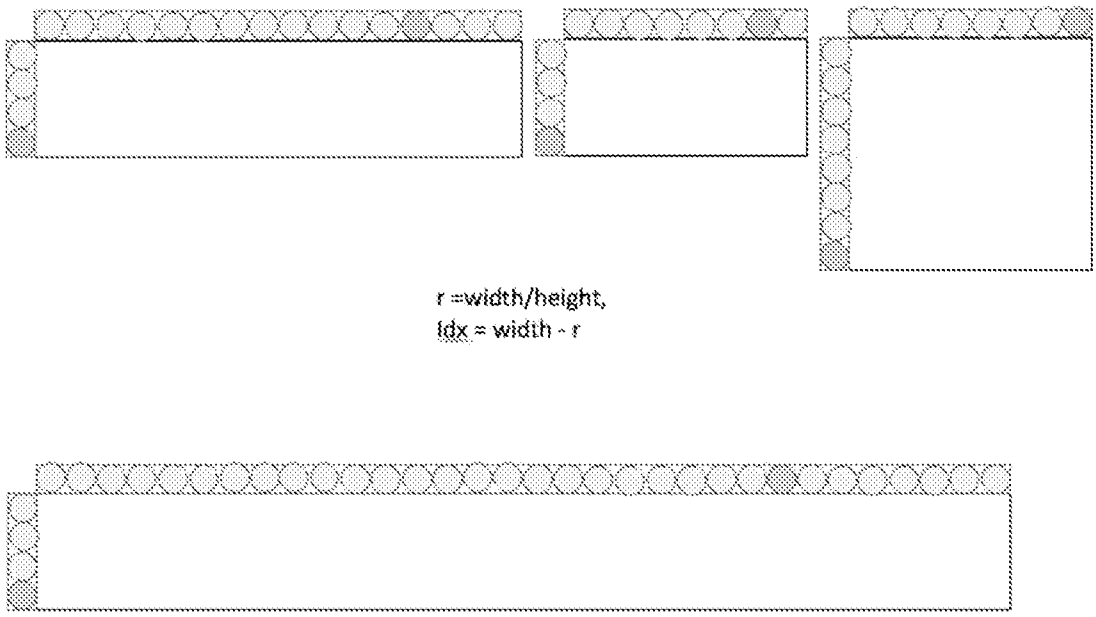
r =width/height,
Idx = width - r

FIG. 9

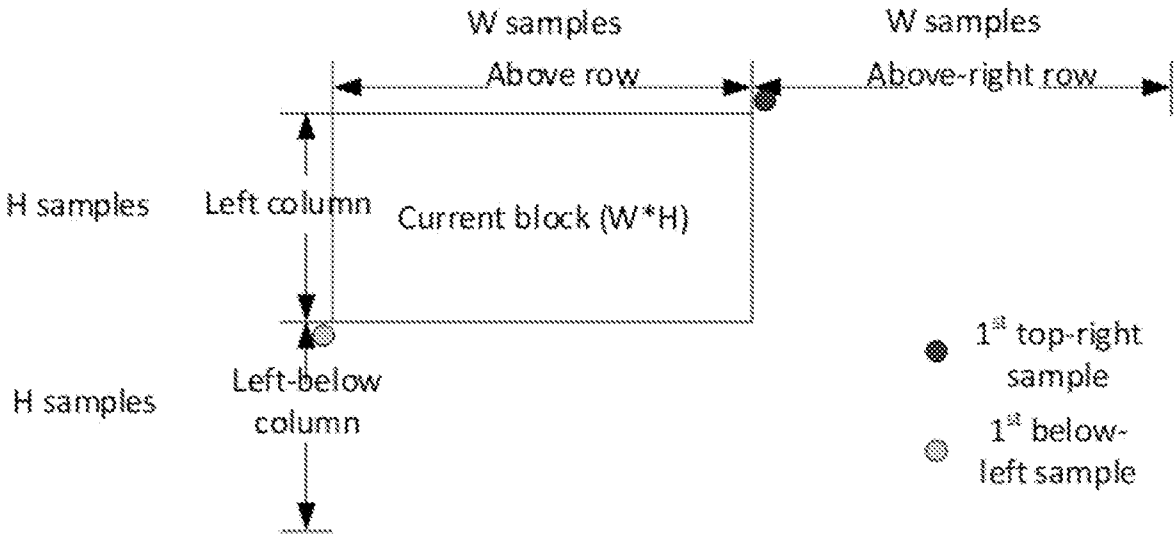
W samples
Above row
W samples
Above-right row
H samples
Left column
Current block (W*H)
H samples
Left-below
column
1st top-right
sample
1st below-
left sample

FIG. 10

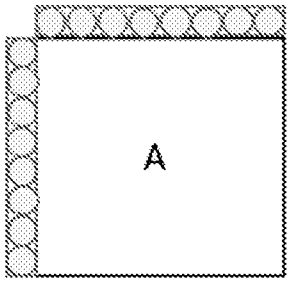

FIG. 11

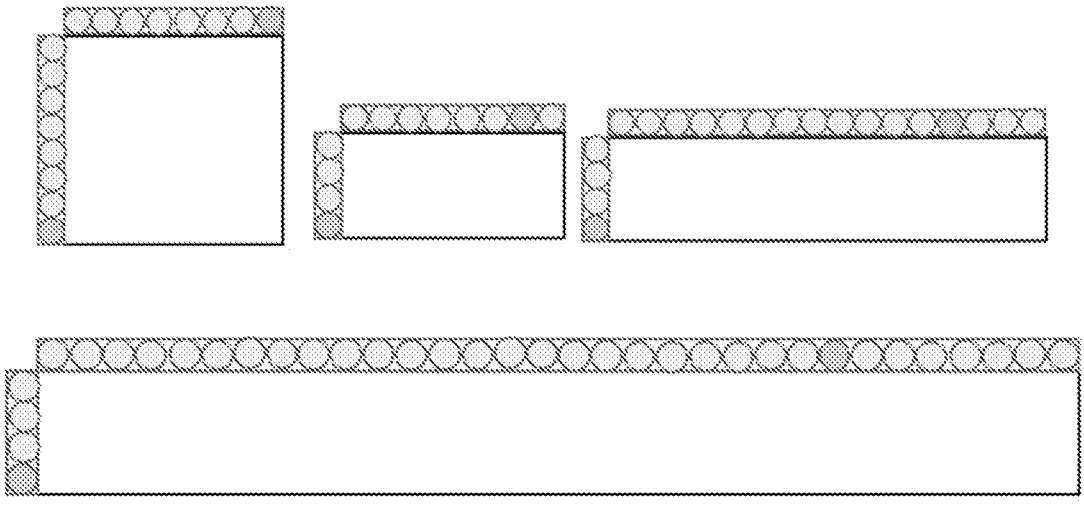

FIG. 12

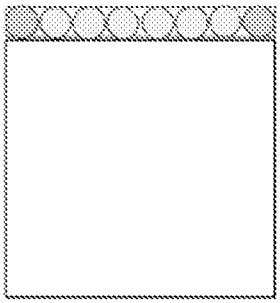

FIG. 13

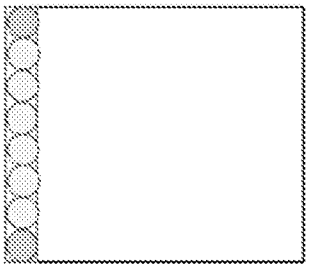

FIG. 14

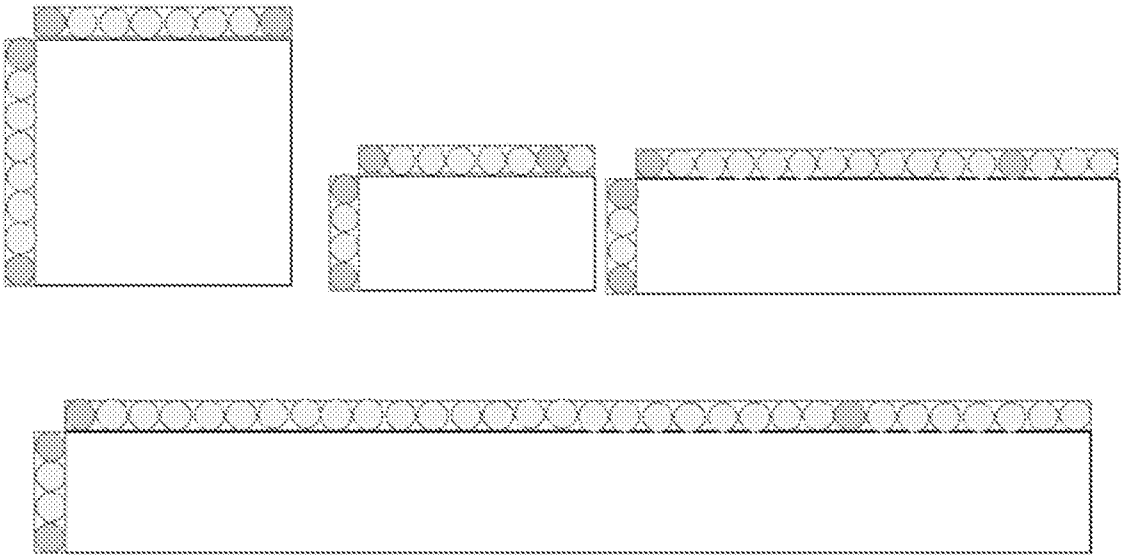

FIG. 15

```
int g_aiMDivTableHighSimp_128[] = {
    65536, 32768, 21845, 16384, 13107, 10922, 9362 , 8192 , 7281 , 6553 , 5957 , 5461 , 5041 , 4681 , 4369 , 4096 ,
    3855 , 3640 , 3449 , 3276 , 3120 , 2978 , 2849 , 2730 , 2621 , 2520 , 2427 , 2340 , 2259 , 2184 , 2114 , 2048 ,
    1985 , 1927 , 1872 , 1820 , 1771 , 1724 , 1680 , 1638 , 1598 , 1560 , 1524 , 1489 , 1456 , 1424 , 1394 , 1365 ,
    1337 , 1310 , 1285 , 1260 , 1236 , 1213 , 1191 , 1170 , 1149 , 1129 , 1110 , 1092 , 1074 , 1057 , 1040 , 1024 ,
    1008 , 992  , 978  , 963  , 949  , 936  , 923  , 910  , 897  , 885  , 873  , 862  , 851  , 840  , 829  , 819  ,
    809  , 799  , 789  , 780  , 771  , 762  , 753  , 744  , 736  , 728  , 720  , 712  , 704  , 697  , 689  , 682  ,
    675  , 668  , 661  , 655  , 648  , 642  , 636  , 630  , 624  , 618  , 612  , 606  , 601  , 595  , 590  , 585  ,
    579  , 574  , 569  , 564  , 560  , 555  , 550  , 546  , 541  , 537  , 532  , 528  , 524  , 520  , 516  , 512  ,
};

int g_aiMDivTableHighSimp_64[] = {
    65536, 32768, 21845, 16384, 13107, 10922, 9362 , 8192 , 7281 , 6553 , 5957 , 5461 , 5041 , 4681 , 4369 , 4096 ,
    3855 , 3640 , 3449 , 3276 , 3120 , 2978 , 2849 , 2730 , 2621 , 2520 , 2427 , 2340 , 2259 , 2184 , 2114 , 2048 ,
    1985 , 1927 , 1872 , 1820 , 1771 , 1724 , 1680 , 1638 , 1598 , 1560 , 1524 , 1489 , 1456 , 1424 , 1394 , 1365 ,
    1337 , 1310 , 1285 , 1260 , 1236 , 1213 , 1191 , 1170 , 1149 , 1129 , 1110 , 1092 , 1074 , 1057 , 1040 , 1024 ,
};

int g_aiMDivTableHighSimp_32[] = {
    65536, 32768, 21845, 16384, 13107, 10922, 9362 , 8192 , 7281 , 6553 , 5957 , 5461 , 5041 , 4681 , 4369 , 4096 ,
    3855 , 3640 , 3449 , 3276 , 3120 , 2978 , 2849 , 2730 , 2621 , 2520 , 2427 , 2340 , 2259 , 2184 , 2114 , 2048 ,
};
```

FIG. 16

```
int iDeltaLuma = maxLuma[0] - minLuma[0];
shift = 0;
add = 0;
if (iDeltaLuma > 64) {
    shift = 4;
    add = 8;
    iDeltaLuma = (iDeltaLuma + add) >> shift;
}
a       = (((maxLuma[1] - minLuma[1]) * g_aiMDivTableHighSimp_64[iDeltaLuma - 1] + add) >> shift);
```

FIG. 17

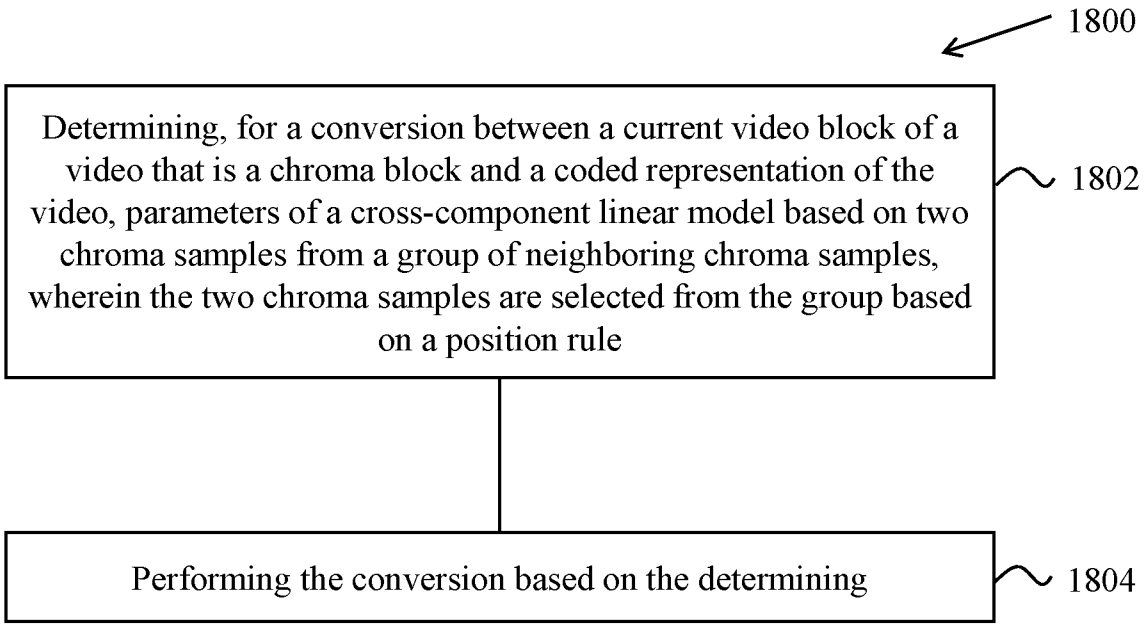
1800
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on two chroma samples from a group of neighboring chroma samples, wherein the two chroma samples are selected from the group based on a position rule
1802
Performing the conversion based on the determining
1804

FIG. 18

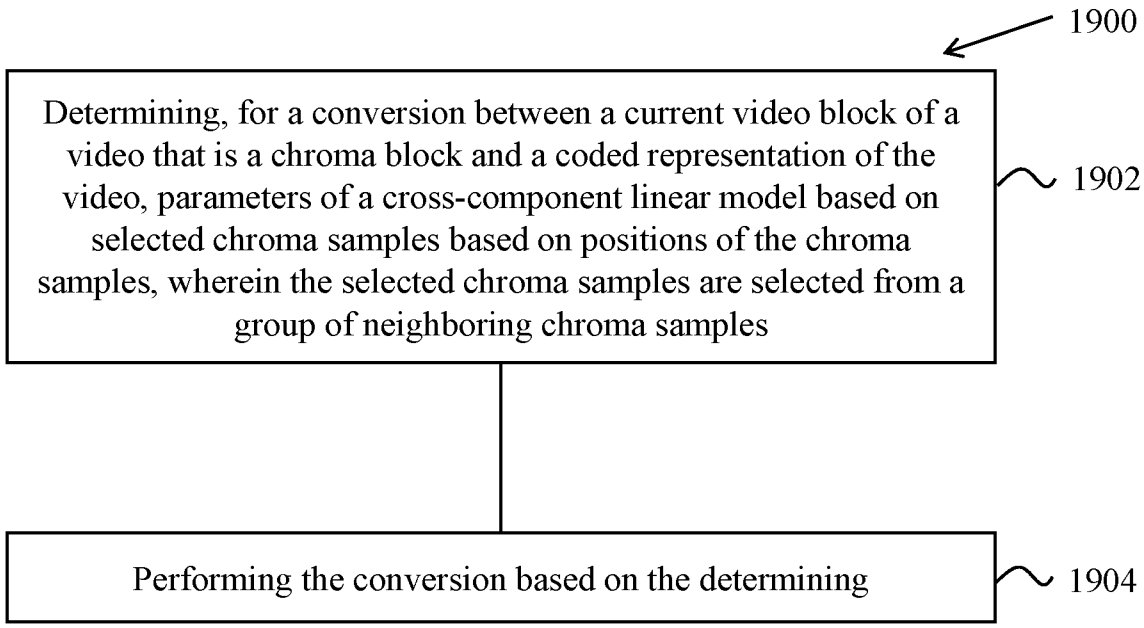
1900
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model based on selected chroma samples based on positions of the chroma samples, wherein the selected chroma samples are selected from a group of neighboring chroma samples
1902
Performing the conversion based on the determining
1904

FIG. 19A

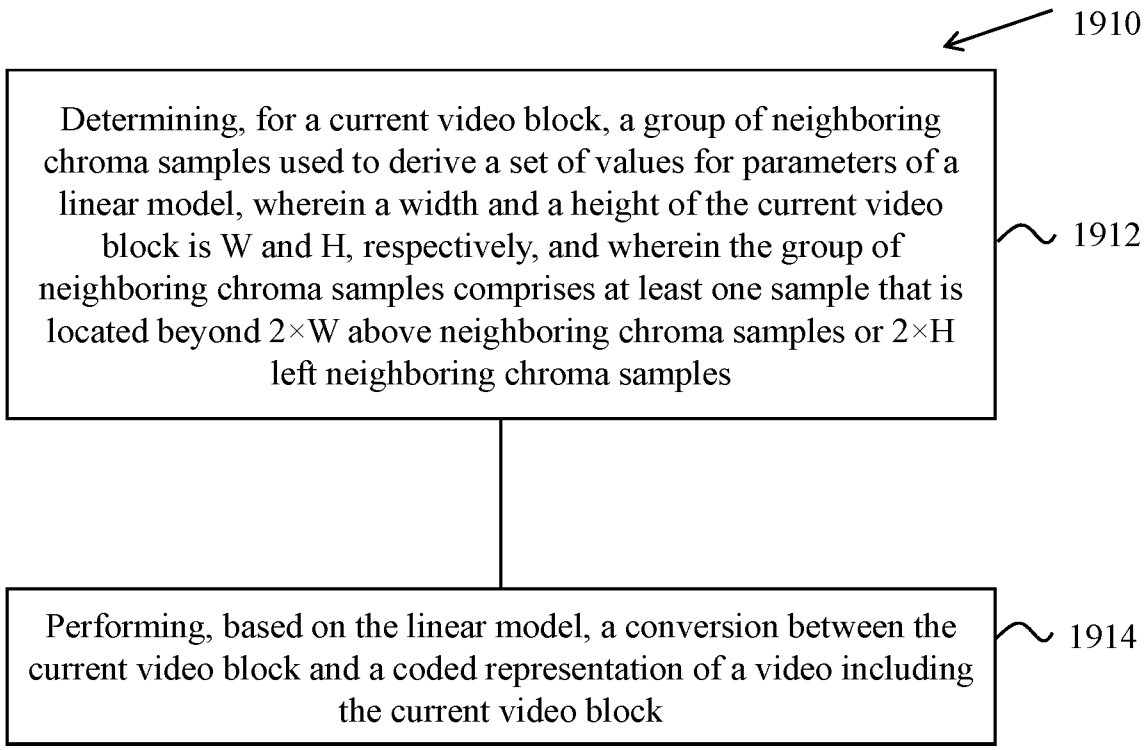
1910
Determining, for a current video block, a group of neighboring chroma samples used to derive a set of values for parameters of a linear model, wherein a width and a height of the current video block is W and H, respectively, and wherein the group of neighboring chroma samples comprises at least one sample that is located beyond 2×W above neighboring chroma samples or 2×H left neighboring chroma samples
1912
Performing, based on the linear model, a conversion between the current video block and a coded representation of a video including the current video block
1914

FIG. 19B

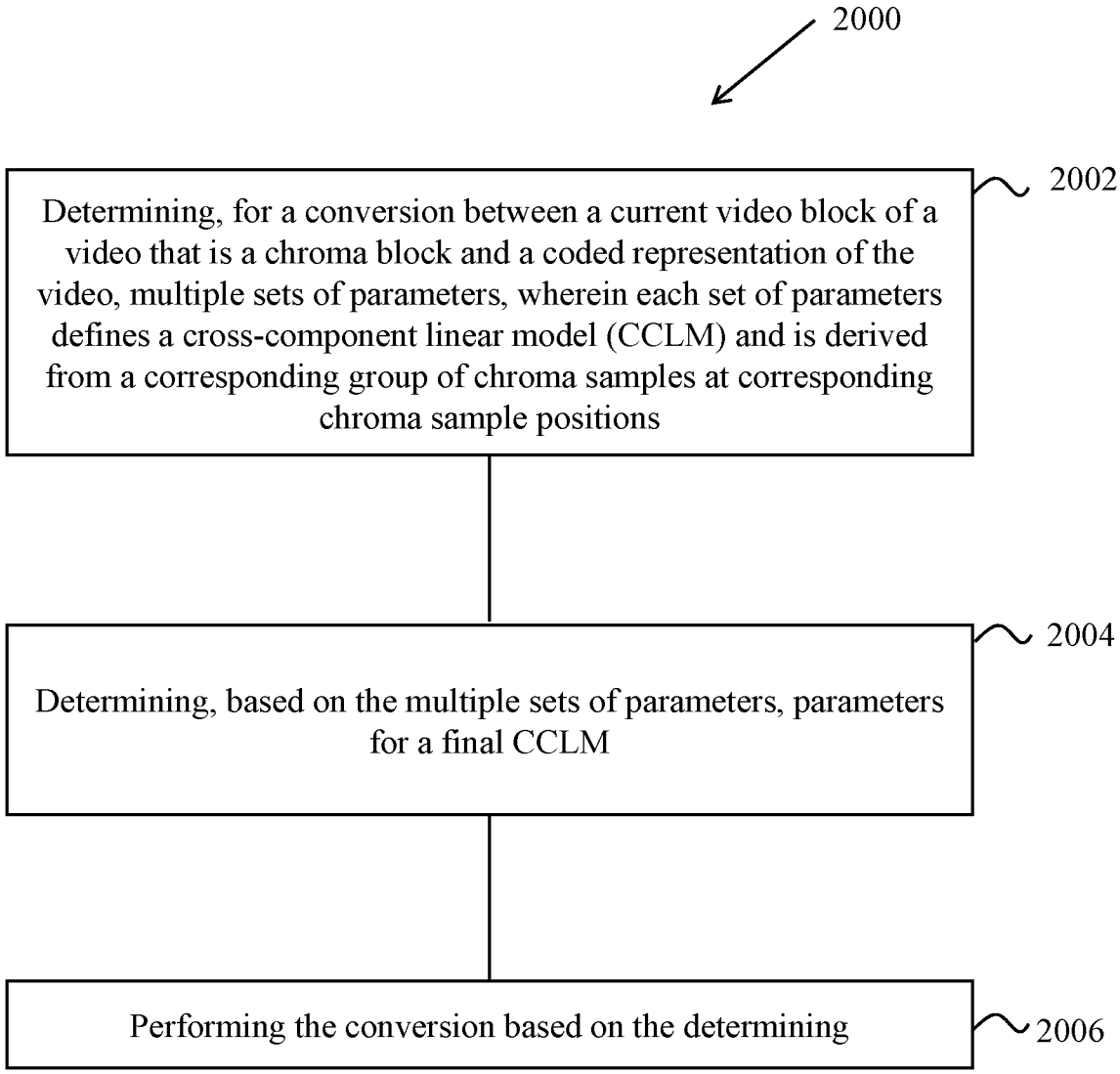
2000
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, multiple sets of parameters, wherein each set of parameters defines a cross-component linear model (CCLM) and is derived from a corresponding group of chroma samples at corresponding chroma sample positions
2002
Determining, based on the multiple sets of parameters, parameters for a final CCLM
2004
Performing the conversion based on the determining
2006

FIG. 20A

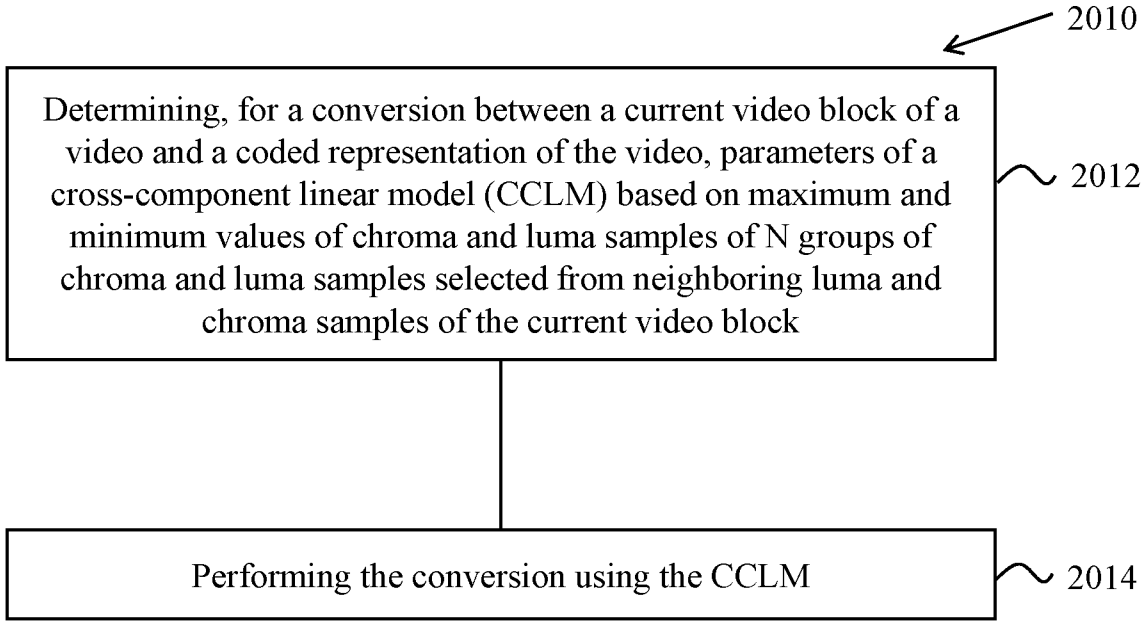
2010
Determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on maximum and minimum values of chroma and luma samples of N groups of chroma and luma samples selected from neighboring luma and chroma samples of the current video block
2012
Performing the conversion using the CCLM
2014

FIG. 20B

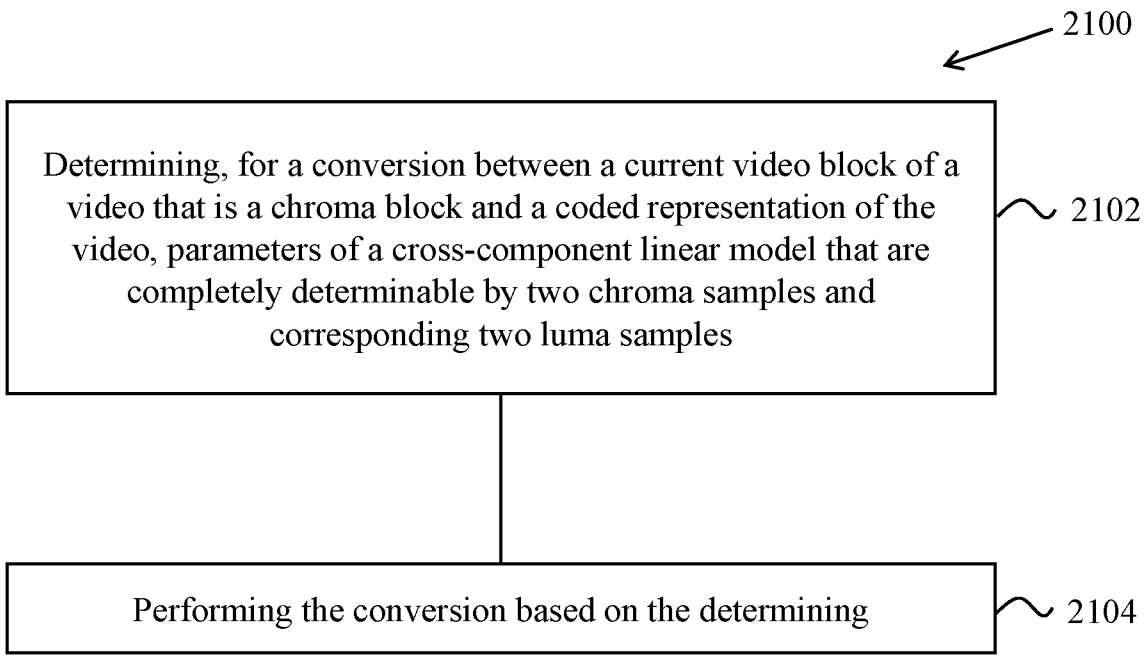
2100
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model that are completely determinable by two chroma samples and corresponding two luma samples
2102
Performing the conversion based on the determining
2104

FIG. 21

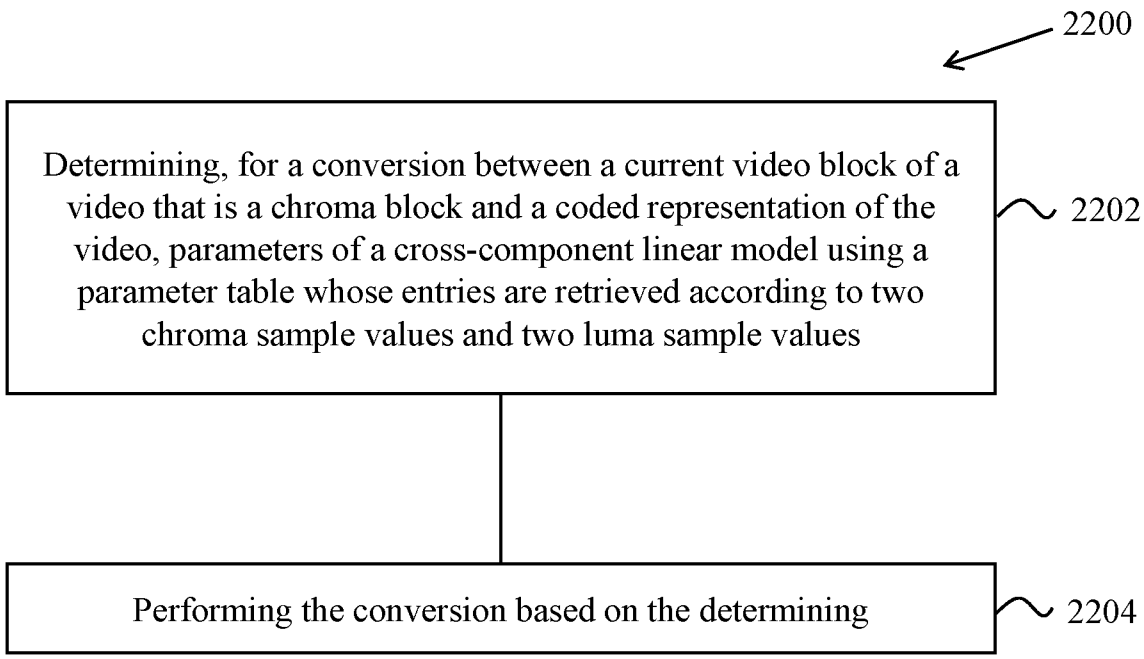
2200
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model using a parameter table whose entries are retrieved according to two chroma sample values and two luma sample values
2202
Performing the conversion based on the determining
2204

FIG. 22

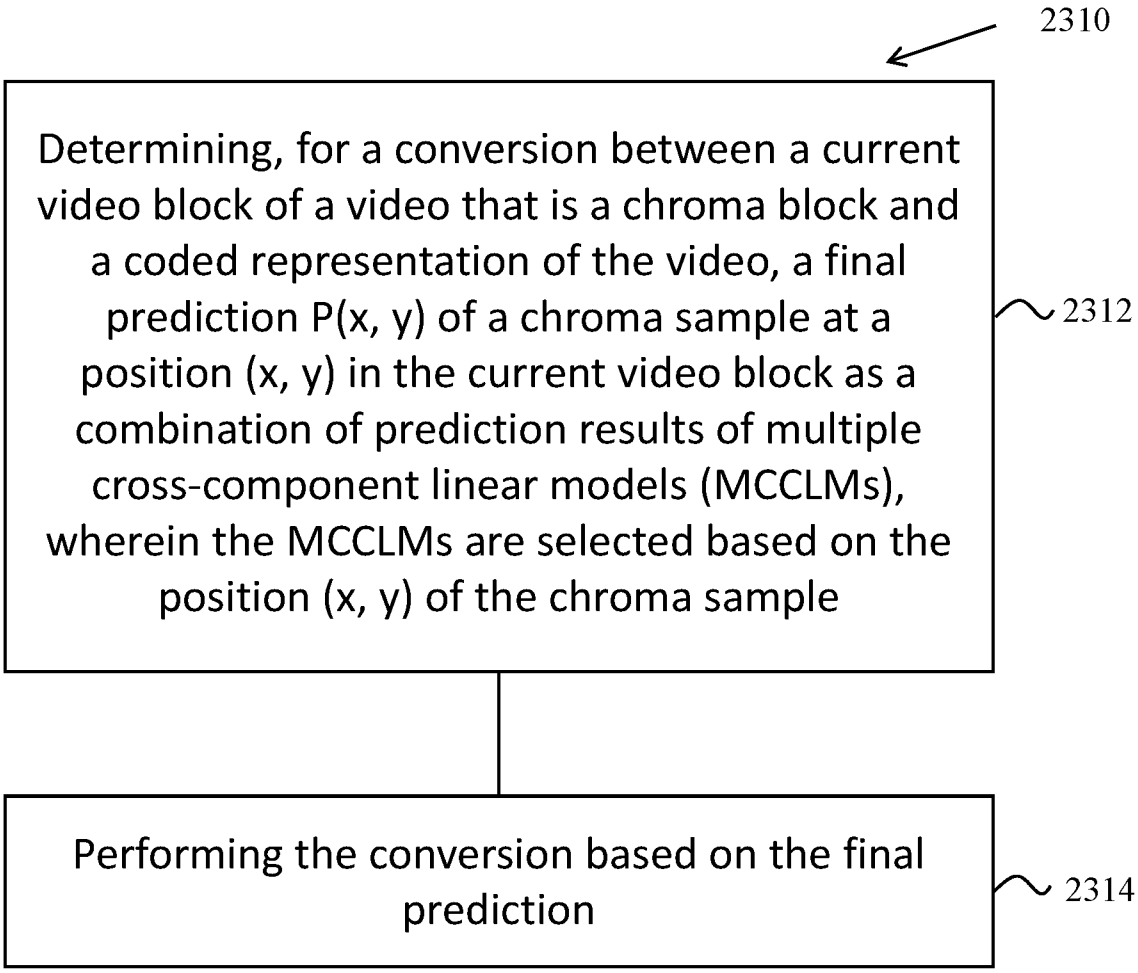
2310
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a final prediction P(x, y) of a chroma sample at a position (x, y) in the current video block as a combination of prediction results of multiple cross-component linear models (MCCLMs), wherein the MCCLMs are selected based on the position (x, y) of the chroma sample
2312
Performing the conversion based on the final prediction
2314

FIG. 23A

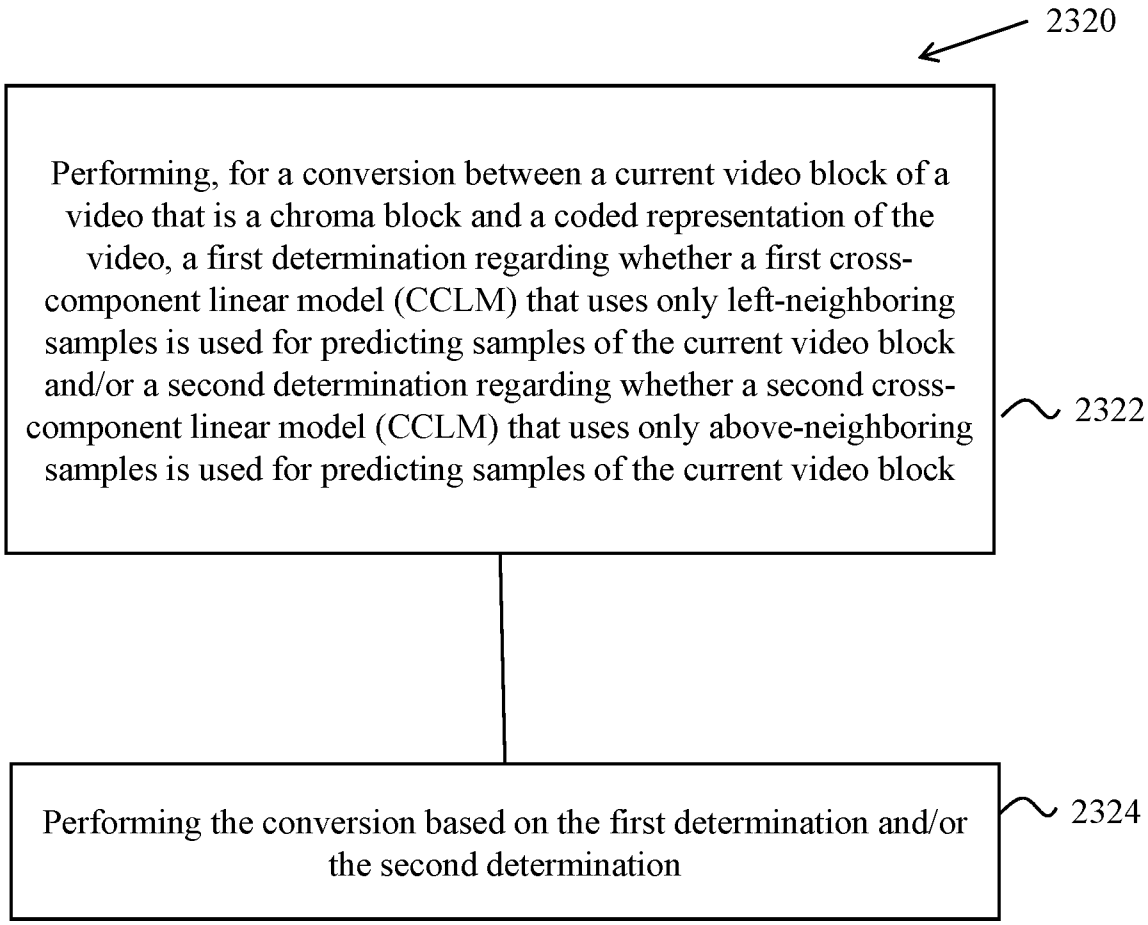
2320
Performing, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a first determination regarding whether a first cross-component linear model (CCLM) that uses only left-neighboring samples is used for predicting samples of the current video block and/or a second determination regarding whether a second cross-component linear model (CCLM) that uses only above-neighboring samples is used for predicting samples of the current video block
2322
Performing the conversion based on the first determination and/or the second determination
2324

FIG. 23B

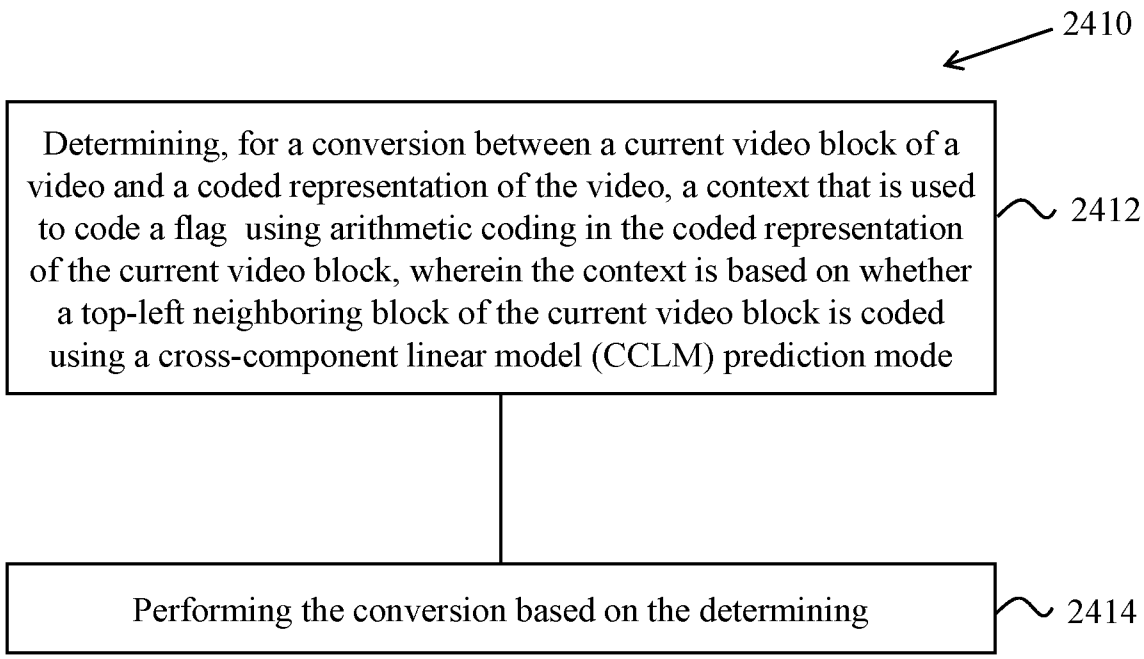
2410
Determining, for a conversion between a current video block of a video and a coded representation of the video, a context that is used to code a flag using arithmetic coding in the coded representation of the current video block, wherein the context is based on whether a top-left neighboring block of the current video block is coded using a cross-component linear model (CCLM) prediction mode
2412
Performing the conversion based on the determining
2414

FIG. 24A

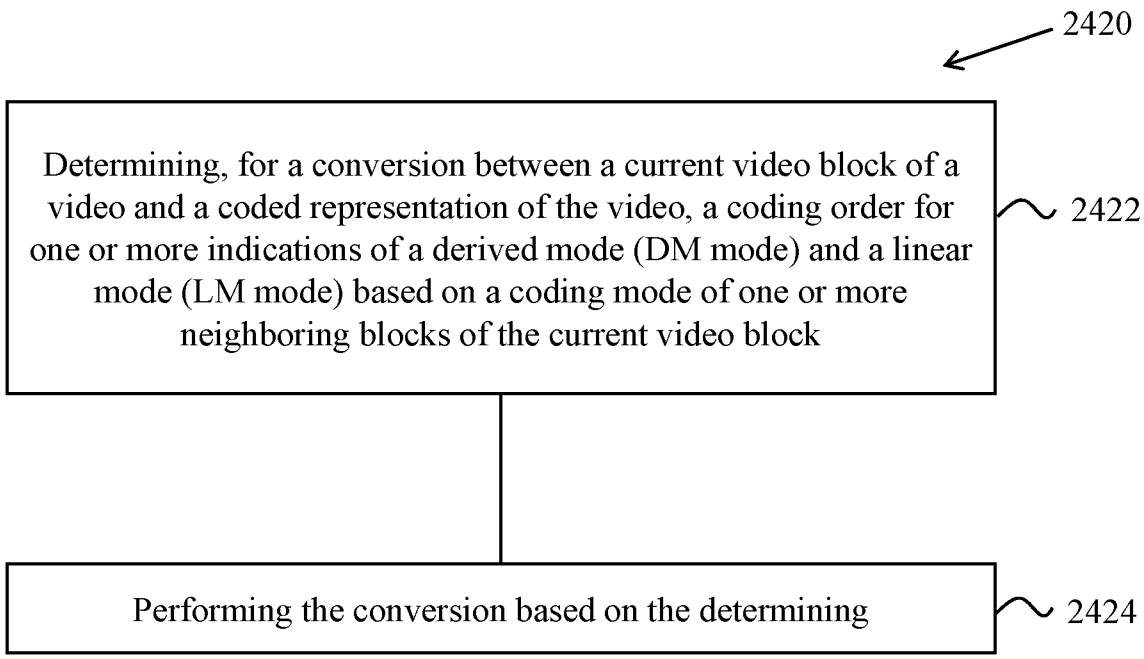
2420
Determining, for a conversion between a current video block of a video and a coded representation of the video, a coding order for one or more indications of a derived mode (DM mode) and a linear mode (LM mode) based on a coding mode of one or more neighboring blocks of the current video block
2422
Performing the conversion based on the determining
2424

FIG. 24B

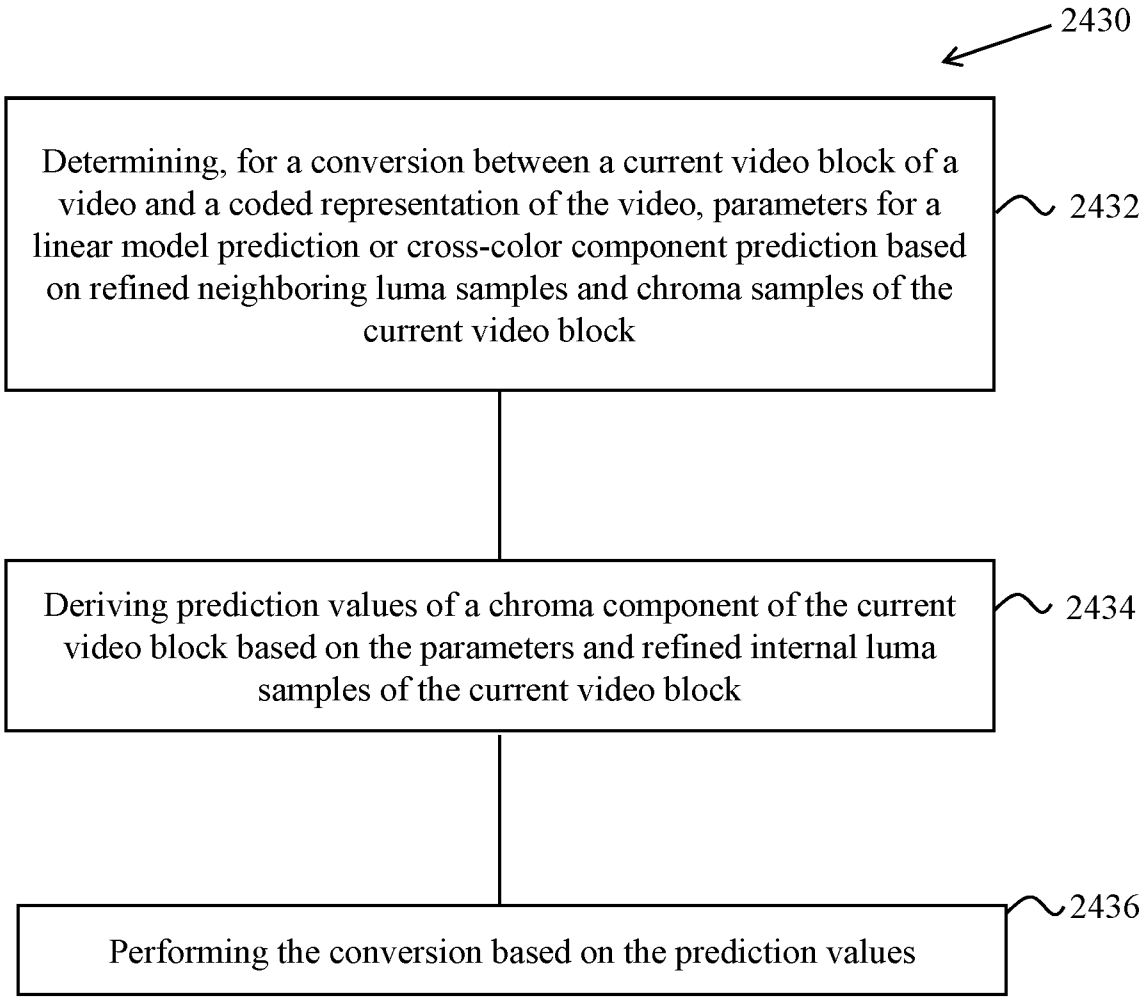
2430
Determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on refined neighboring luma samples and chroma samples of the current video block
2432
Deriving prediction values of a chroma component of the current video block based on the parameters and refined internal luma samples of the current video block
2434
Performing the conversion based on the prediction values
2436

FIG. 24C

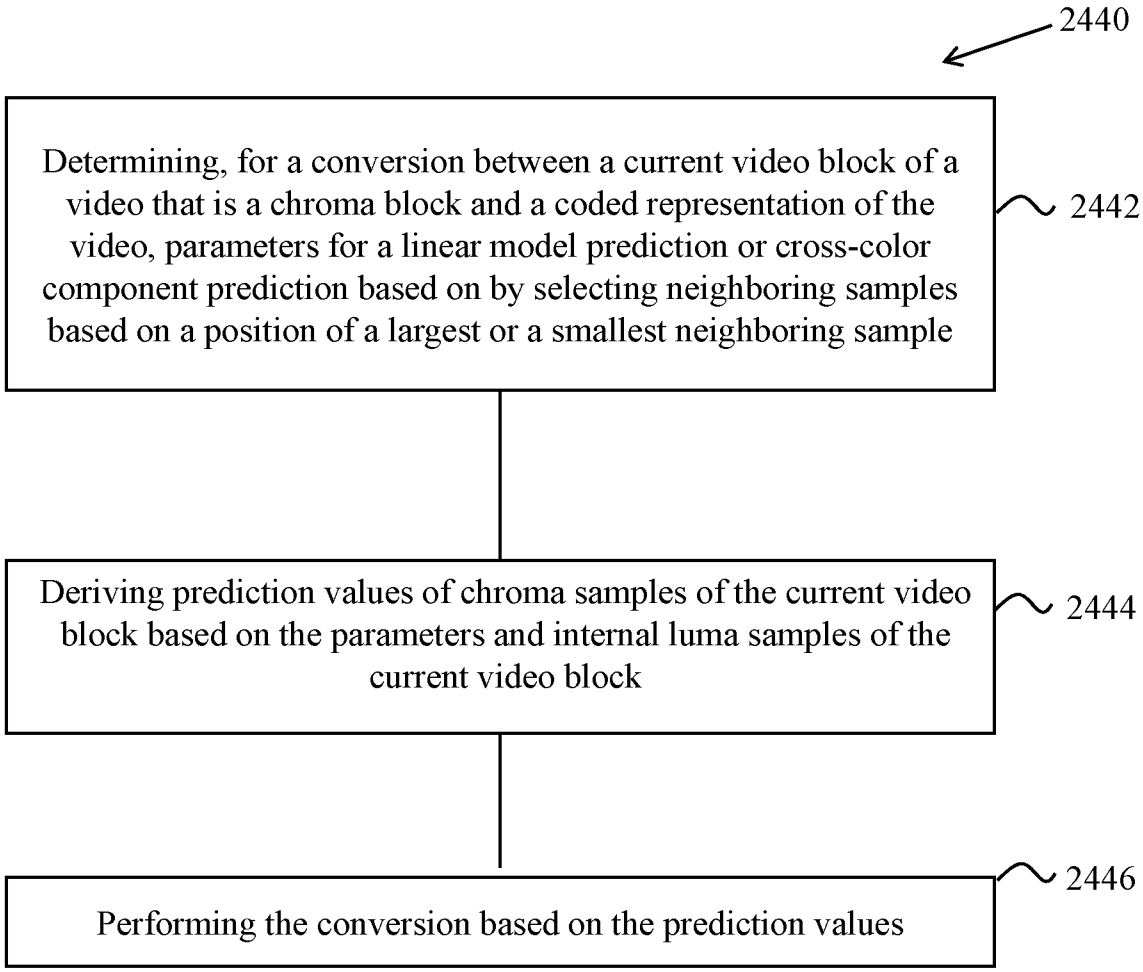
2440
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on by selecting neighboring samples based on a position of a largest or a smallest neighboring sample
2442
Deriving prediction values of chroma samples of the current video block based on the parameters and internal luma samples of the current video block
2444
Performing the conversion based on the prediction values
2446

FIG. 24D

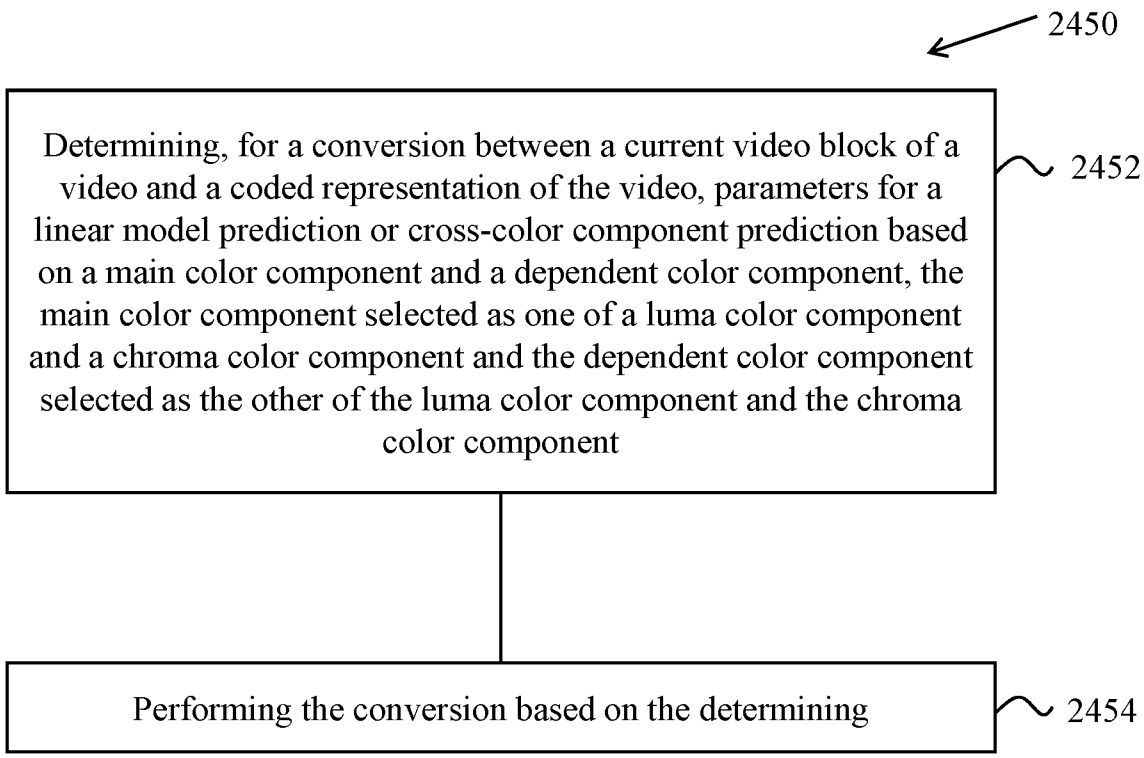
2450
Determining, for a conversion between a current video block of a video and a coded representation of the video, parameters for a linear model prediction or cross-color component prediction based on a main color component and a dependent color component, the main color component selected as one of a luma color component and a chroma color component and the dependent color component selected as the other of the luma color component and the chroma color component
2452
Performing the conversion based on the determining
2454

FIG. 24E

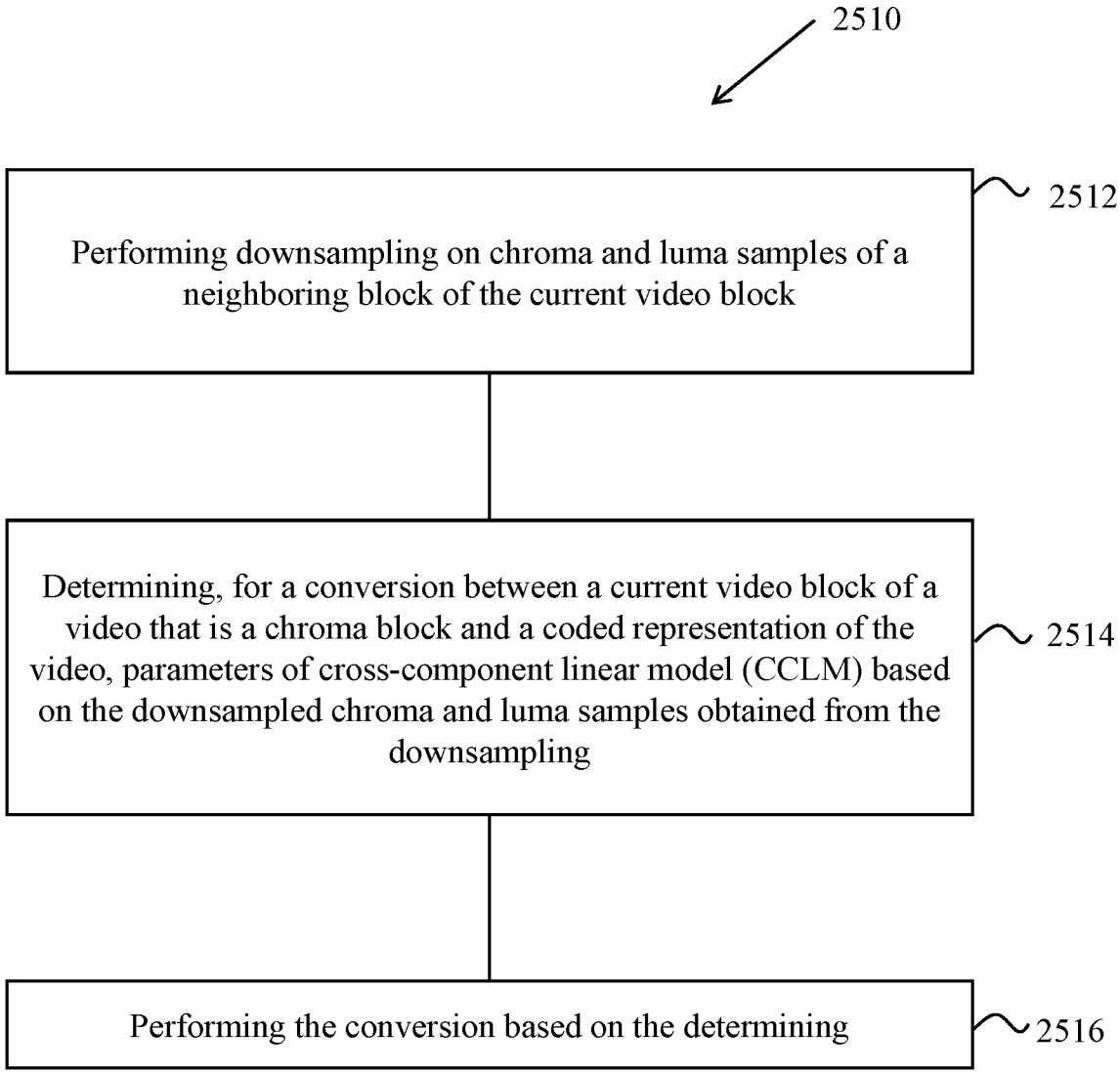
2510
Performing downsampling on chroma and luma samples of a neighboring block of the current video block
2512
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on the downsampled chroma and luma samples obtained from the downsampling
2514
Performing the conversion based on the determining
2516

FIG. 25A

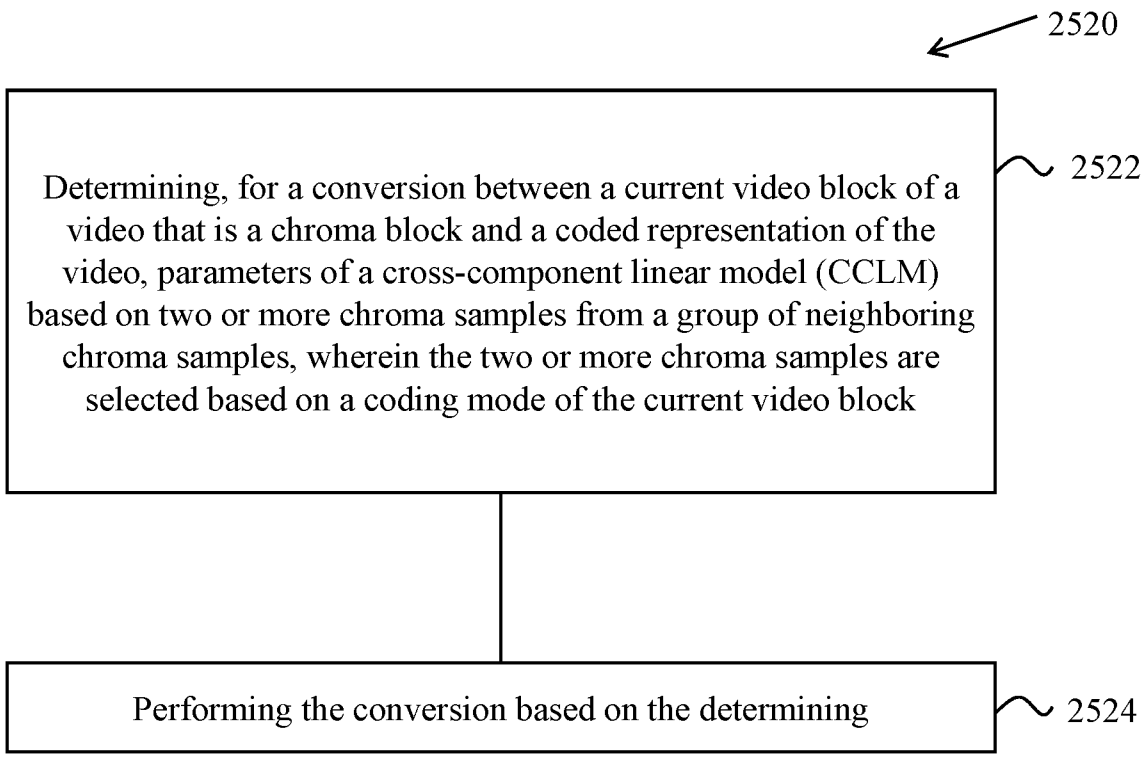
2520
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or more chroma samples from a group of neighboring chroma samples, wherein the two or more chroma samples are selected based on a coding mode of the current video block
2522
Performing the conversion based on the determining
2524

FIG. 25B

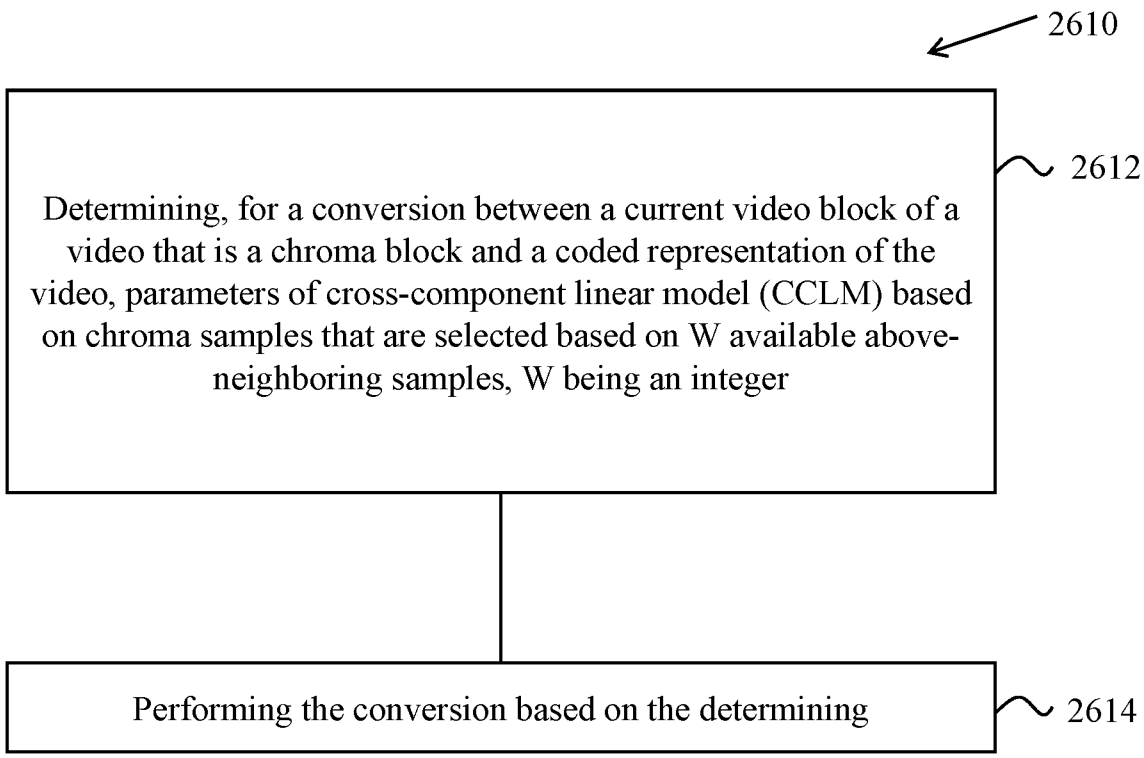
2610
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on W available above-neighboring samples, W being an integer
2612
Performing the conversion based on the determining
2614

FIG. 26A

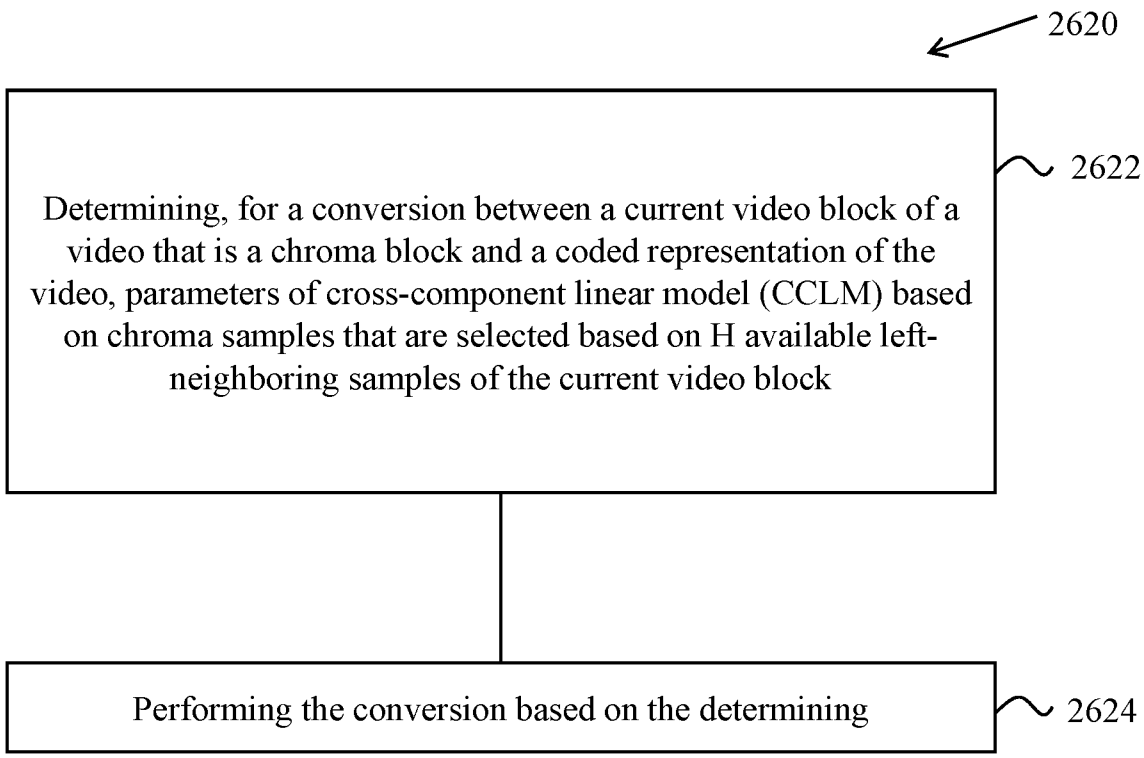
2620
2622
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of cross-component linear model (CCLM) based on chroma samples that are selected based on H available left-neighboring samples of the current video block
Performing the conversion based on the determining
2624

FIG. 26B

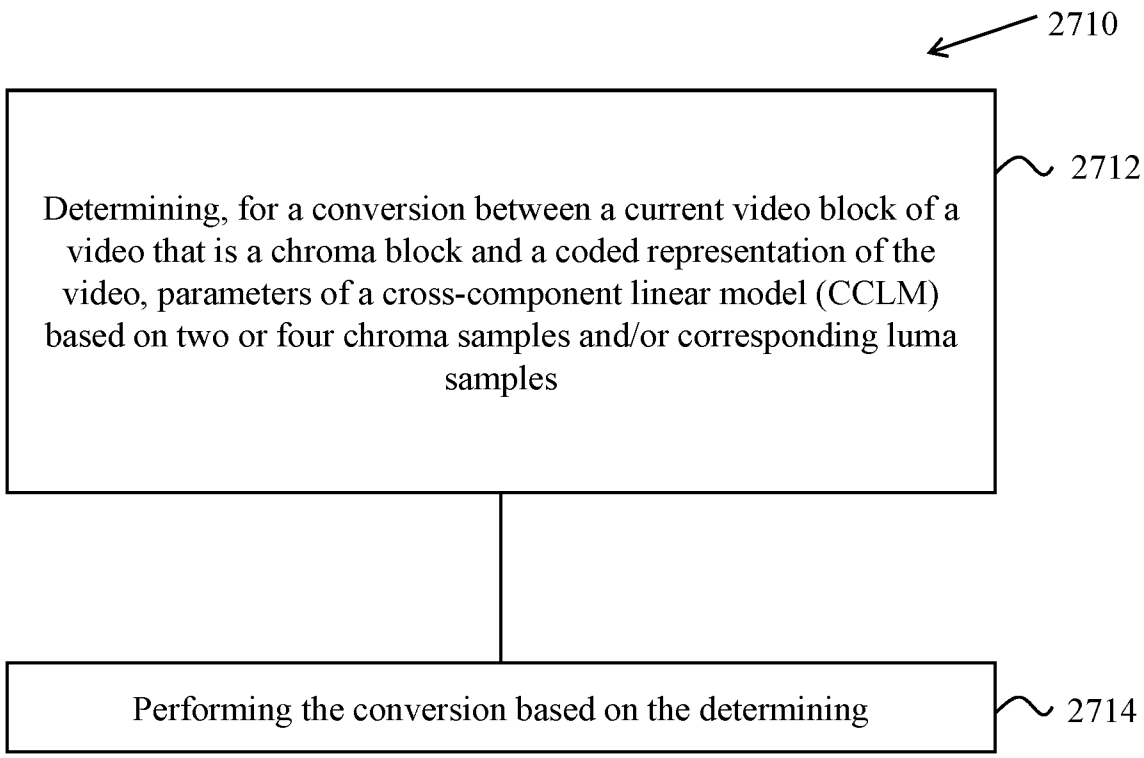
2710
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on two or four chroma samples and/or corresponding luma samples
2712
Performing the conversion based on the determining
2714

FIG. 27A

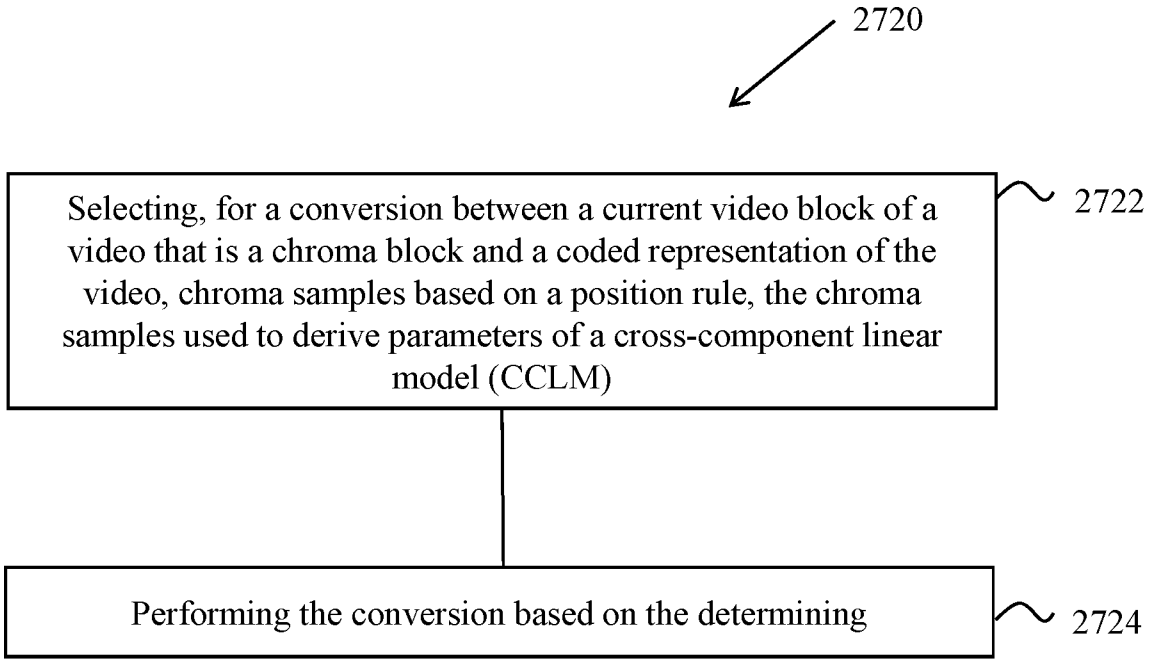
2720
Selecting, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, chroma samples based on a position rule, the chroma samples used to derive parameters of a cross-component linear model (CCLM)
2722
Performing the conversion based on the determining
2724

FIG. 27B

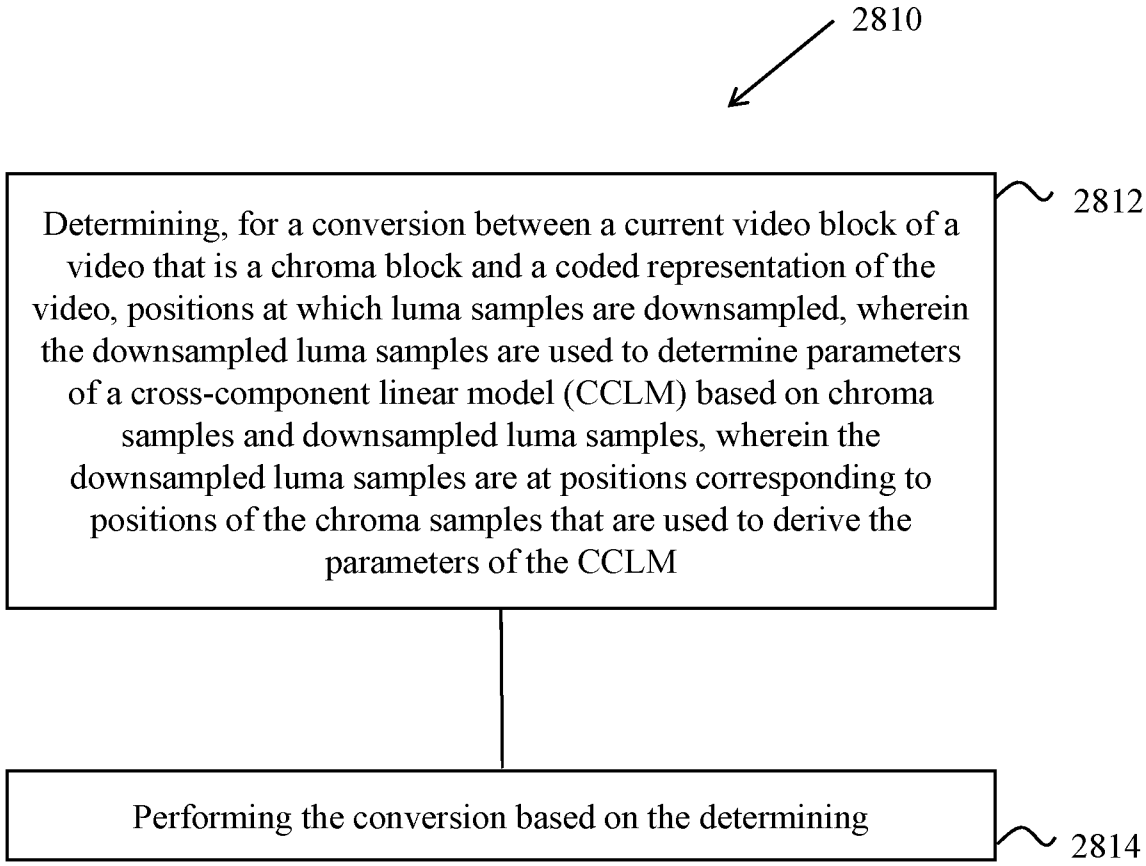
2810
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, positions at which luma samples are downsampled, wherein the downsampled luma samples are used to determine parameters of a cross-component linear model (CCLM) based on chroma samples and downsampled luma samples, wherein the downsampled luma samples are at positions corresponding to positions of the chroma samples that are used to derive the parameters of the CCLM
2812
Performing the conversion based on the determining
2814

FIG. 28A

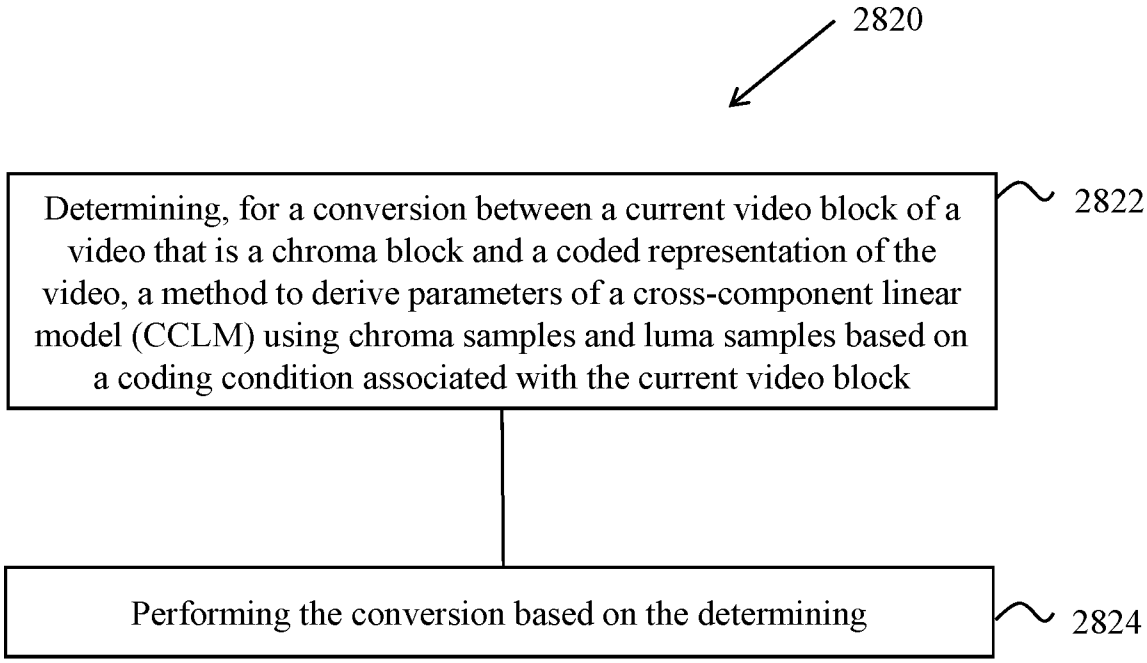
2820
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, a method to derive parameters of a cross-component linear model (CCLM) using chroma samples and luma samples based on a coding condition associated with the current video block
2822
Performing the conversion based on the determining
2824

FIG. 28B

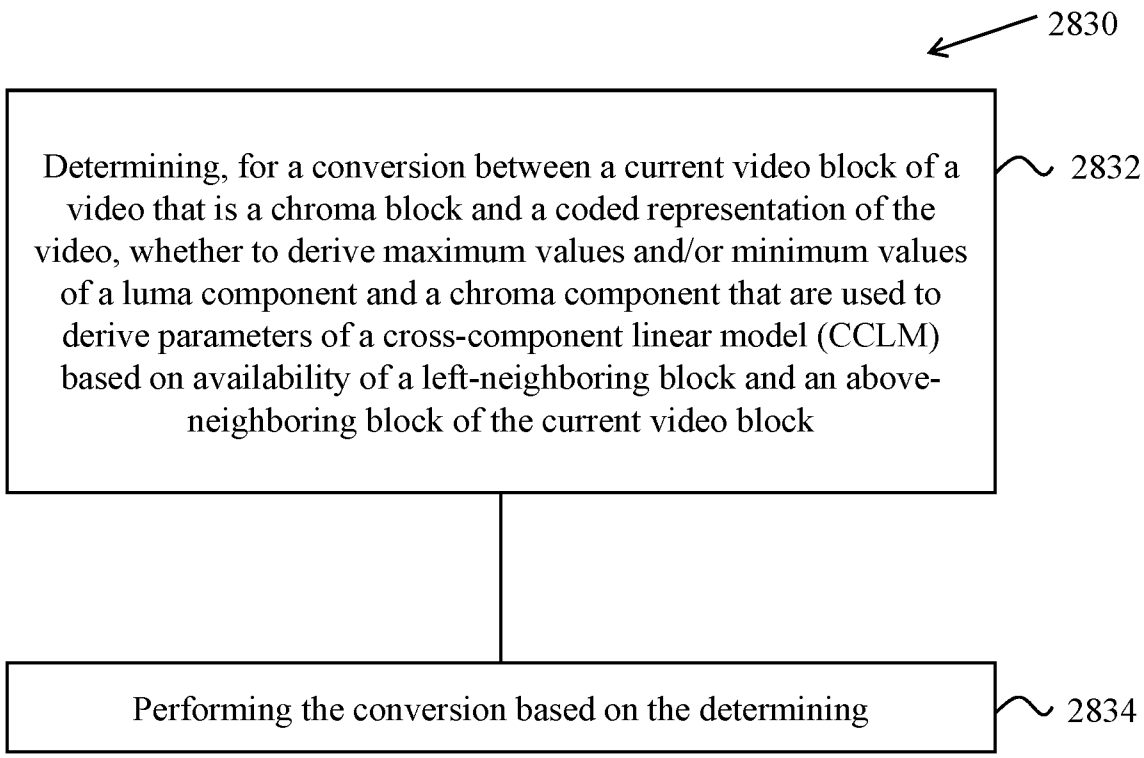
2830
Determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, whether to derive maximum values and/or minimum values of a luma component and a chroma component that are used to derive parameters of a cross-component linear model (CCLM) based on availability of a left-neighboring block and an above-neighboring block of the current video block
2832
Performing the conversion based on the determining
2834

FIG. 28C

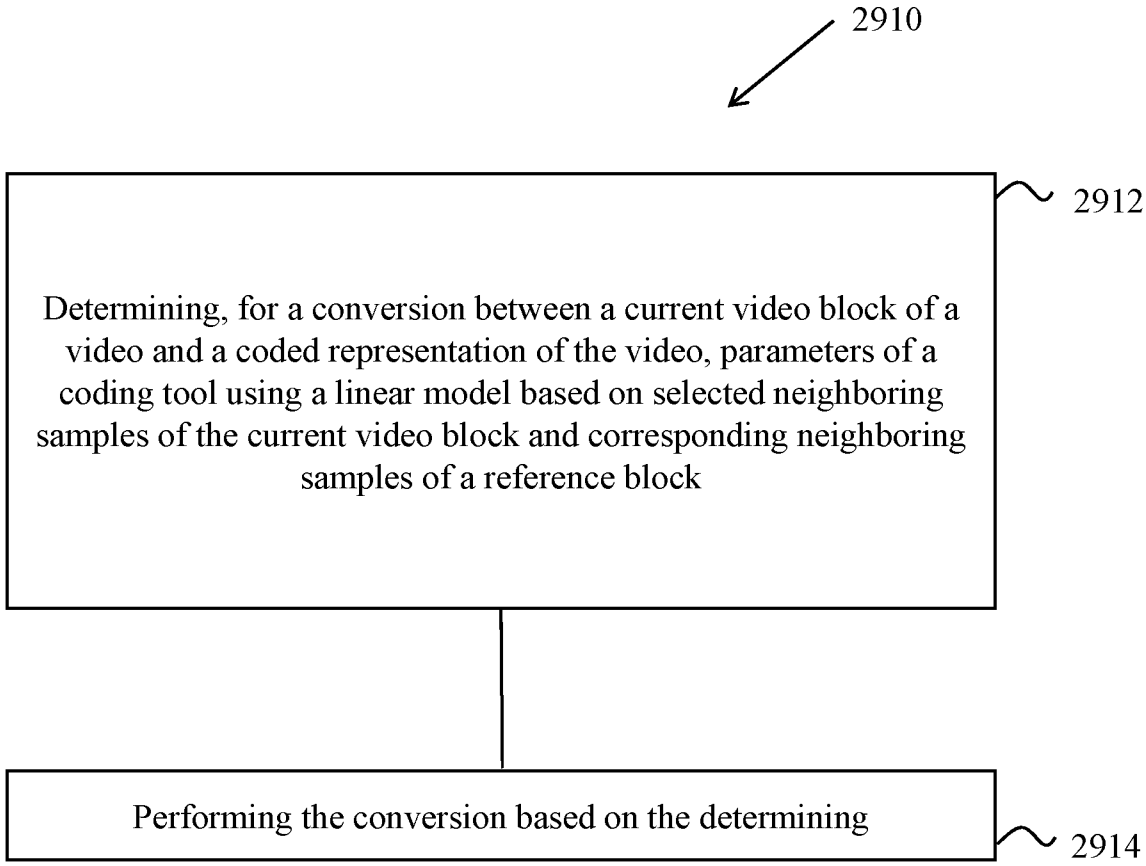
2910
2912
Determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a coding tool using a linear model based on selected neighboring samples of the current video block and corresponding neighboring samples of a reference block
Performing the conversion based on the determining
2914

FIG. 29A

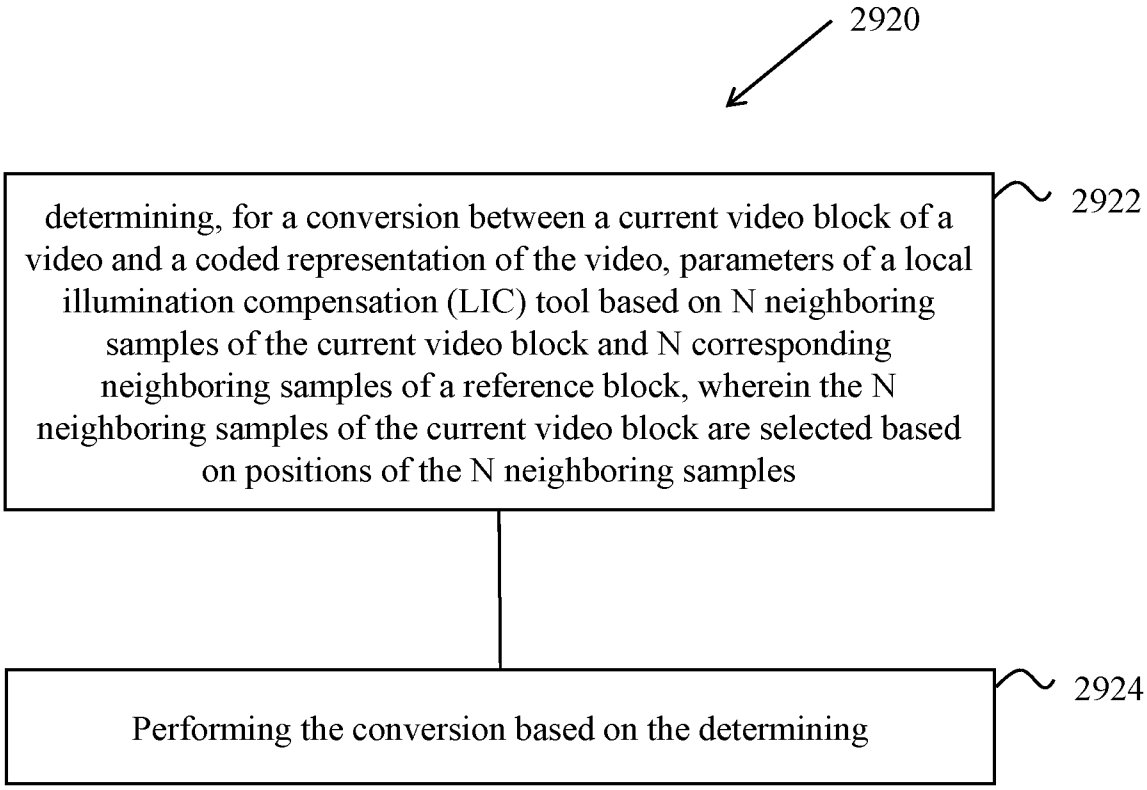
2920
determining, for a conversion between a current video block of a video and a coded representation of the video, parameters of a local illumination compensation (LIC) tool based on N neighboring samples of the current video block and N corresponding neighboring samples of a reference block, wherein the N neighboring samples of the current video block are selected based on positions of the N neighboring samples
2922
Performing the conversion based on the determining
2924

FIG. 29B

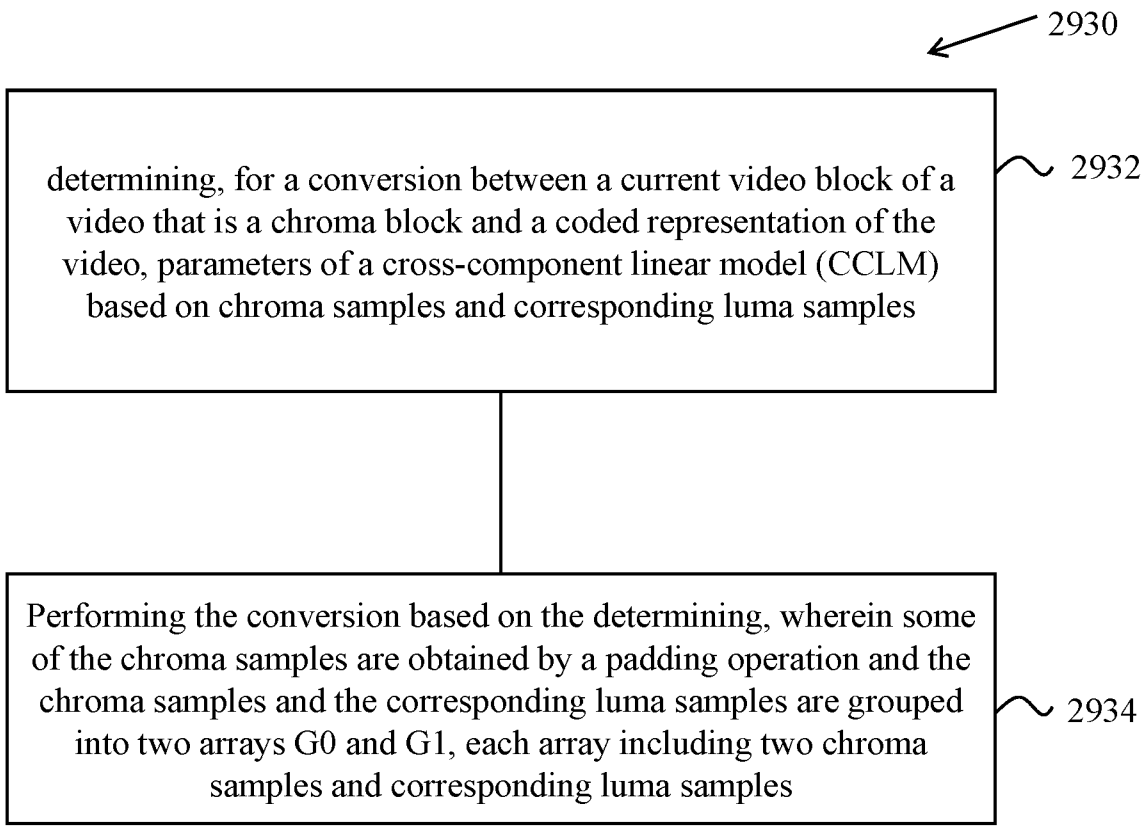
2930
determining, for a conversion between a current video block of a video that is a chroma block and a coded representation of the video, parameters of a cross-component linear model (CCLM) based on chroma samples and corresponding luma samples
2932
Performing the conversion based on the determining, wherein some of the chroma samples are obtained by a padding operation and the chroma samples and the corresponding luma samples are grouped into two arrays G0 and G1, each array including two chroma samples and corresponding luma samples
2934

FIG. 29C

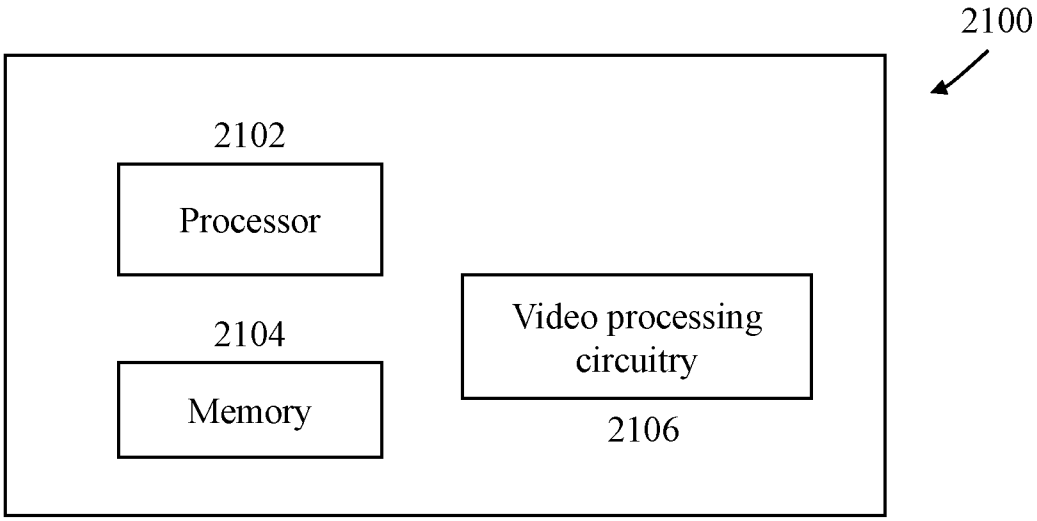
2100
2102
Processor
2104
Memory
Video processing circuitry
2106

FIG. 30A

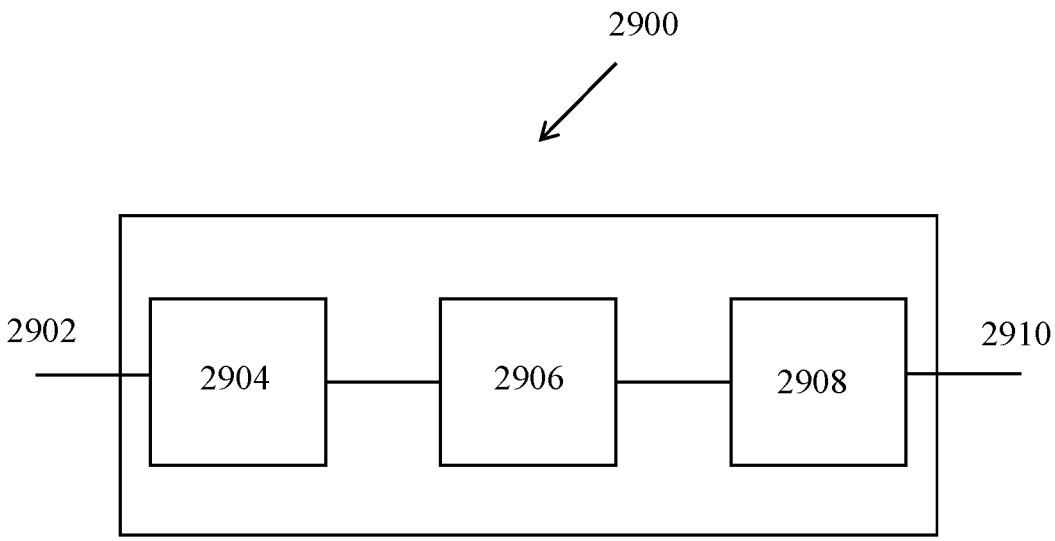

FIG. 30B

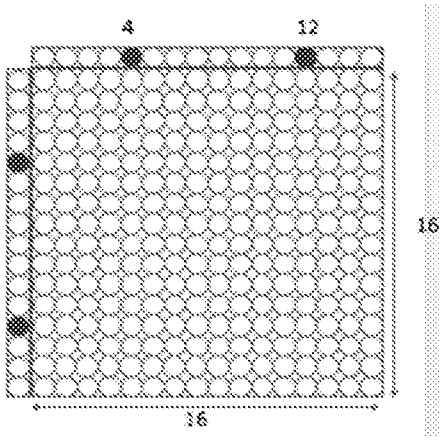

FIG. 31A

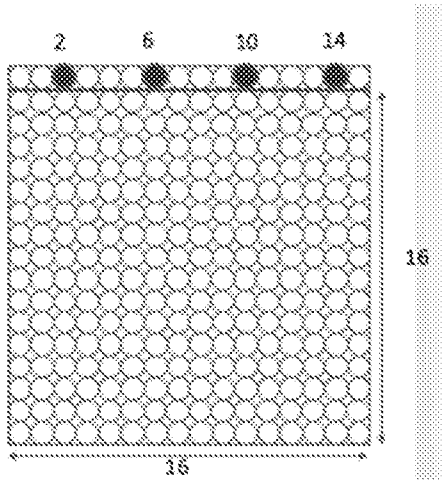

FIG. 31B

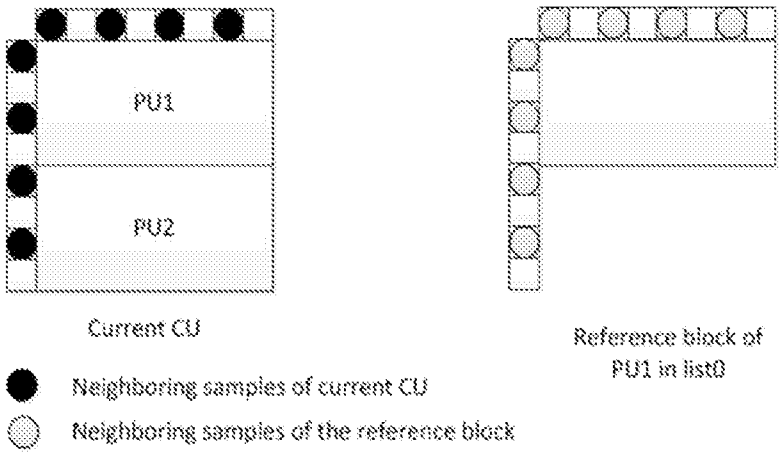
PU1
PU2
Current CU
Reference block of PU1 in list0
Neighboring samples of current CU
Neighboring samples of the reference block

FIG. 32

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2019/121850

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/117(2014.01)i; H04N 19/159(2014.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNPAT,CNKI,EPODOC,WPI,IEEE:conversion,video,context,code,block,neighboring,cross component linear model,CCLM, prediction,chroma,derived mode,DM,LM,linear mode

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2018140587 A1 (QUALCOMM INC.) 02 August 2018 (2018-08-02) description, paragraph 117	1-6,31-34
X	US 2013128966 A1 (FUTUREWEI TECHNOLOGIES INC.) 23 May 2013 (2013-05-23) description, paragraph 24	7-11,31-34
X	US 2015365684 A1 (MEDIATEK INC.) 17 December 2015 (2015-12-17) description, paragraphs 22, 26	12-34
A	WO 2016115708 A1 (MEDIATEK SINGAPORE PTE. LTD.) 28 July 2016 (2016-07-28) the whole document	1-34
A	US 2018063531 A1 (QUALCOMM INC.) 01 March 2018 (2018-03-01) the whole document	1-34
A	CN 104871537 A (MEDIATEK INC.) 26 August 2015 (2015-08-26) the whole document	1-34
A	CHEN, jianle. "Algorithm Description of Joint Exploration Test Model 4" *Joint Video Exploration Team (JVET),* 21 October 2016 (2016-10-21), the whole document	1-34

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
19 January 2020	**07 February 2020**
Name and mailing address of the ISA/CN **National Intellectual Property Administration, PRC** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing** **100088** **China** Facsimile No. **(86-10)62019451**	Authorized officer **XI,Huining** Telephone No. **86-10-53961813**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2019/121850

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
WO	2018140587	A1	02 August 2018	CN	110169064	A	23 August 2019
				SG	11201905243Y	A	27 August 2019
				KR	20190110548	A	30 September 2019
				AU	2018212665	A1	04 July 2019
				EP	3574650	A1	04 December 2019
				TW	201832562	A	01 September 2018
				US	2018220130	A1	02 August 2018
US	2013128966	A1	23 May 2013	EP	2781097	B1	10 January 2018
				CN	103959790	B	24 May 2017
				CN	103959790	A	30 July 2014
				WO	2013071889	A1	23 May 2013
				US	9344722	B2	17 May 2016
				EP	2781097	A4	06 May 2015
				EP	2781097	A1	24 September 2014
US	2015365684	A1	17 December 2015	EP	2920964	B1	09 May 2018
				CA	2896637	A1	02 October 2014
				US	9736487	B2	15 August 2017
				WO	2014154094	A1	02 October 2014
				US	2017295366	A1	12 October 2017
				US	10154268	B2	11 December 2018
				CN	104871537	B	16 March 2018
				EP	2920964	A1	23 September 2015
				CN	104871537	A	26 August 2015
				EP	2920964	A4	13 April 2016
				CA	2896637	C	13 February 2018
WO	2016115708	A1	28 July 2016		None		
US	2018063531	A1	01 March 2018	WO	2018039596	A1	01 March 2018
				EP	3504877	A1	03 July 2019
				CN	109792538	A	21 May 2019
				US	10390015	B2	20 August 2019
CN	104871537	A	26 August 2015	EP	2920964	B1	09 May 2018
				CA	2896637	A1	02 October 2014
				US	9736487	B2	15 August 2017
				WO	2014154094	A1	02 October 2014
				US	2017295366	A1	12 October 2017
				US	10154268	B2	11 December 2018
				CN	104871537	B	16 March 2018
				EP	2920964	A1	23 September 2015
				US	2015365684	A1	17 December 2015
				EP	2920964	A4	13 April 2016
				CA	2896637	C	13 February 2018