CODING METHOD, BITSTREAM, ENCODER, DECODER, AND STORAGE MEDIUM

TECHNICAL FIELD

[0001] Embodiments of the disclosure relate to the field of video coding technology, and in particular, to a coding method, a bitstream, an encoder, a decoder, and a storage medium.

BACKGROUND

[0002] As demands for video display quality increase, high-resolution videos such as high-definition videos and ultra-high-definition videos have emerged. However, a high-resolution video typically contains more information and therefore requires more bandwidth. To reduce bandwidth requirements, video coding standards relating to video compression have been introduced.

[0003] In video coding standards, in common intra prediction methods such as intra block copy (IBC), decoder-side intra mode derivation (DIMD), and template-based intra mode derivation (TIMD), block-vector-based prediction can be derived by constructing a block vector list. However, current approaches for constructing a block vector list are relatively fixed and are poor in flexibility, and thus are poorly adaptable to different prediction scenarios, which degrades intra prediction efficiency and thus affects coding performance.

SUMMARY

[0004] Embodiments of the disclosure provide a coding method, a bitstream, an encoder, a decoder, and a storage medium, which can improve intra prediction efficiency and improve coding performance.

[0005] Technical solutions of embodiments of the disclosure can be implemented as follows.

[0006] According to a first aspect, a decoding method is provided in embodiments of the disclosure. The method is applied to a decoder. The method includes the following. A

bitstream is parsed to determine first syntax flag information. If the first syntax flag information indicates that a preset intra prediction mode is to be applied to a current block, a block vector list for the current block is constructed. A prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample precision block vector, where the subsample precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM). The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0007] According to a second aspect, an encoding method is provided in embodiments of the disclosure. The method is applied to an encoder. The method includes the following. If it is determined that a preset intra prediction mode is to be applied to a current block, first syntax flag information is signalled into a bitstream, where the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block. A block vector list for the current block is constructed. A prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample precision block vector, where the subsample precision block vector is used for DIMD, or TIMD, or SGPM. The block

vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0008] According to a third aspect, a bitstream is provided in embodiments of the disclosure. The bitstream is generated by encoding information to be encoded; where the information to be encoded includes at least one of: first syntax flag information, second syntax flag information, or third syntax flag information.

[0009] According to a fourth aspect, an encoder is provided in embodiments of the disclosure. The encoder includes a first determining unit. The first determining unit is configured to: signal first syntax flag information into a bitstream if it is determined that a preset intra prediction mode is to be applied to a current block, where the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block; construct a block vector list for the current block; and determine a prediction block for the current block according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample precision block vector, where the subsample precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0010] According to a fifth aspect, an encoder is provided in embodiments of the disclosure. The encoder includes a first memory and a first processor. The first memory is configured to store computer programs executable by the first processor. The first processor is configured to, when executing the computer programs, perform the method described in the second aspect.

[0011] According to a sixth aspect, a decoder is provided in embodiments of the disclosure. The decoder includes a second determining unit. The second determining unit is configured to: decode a bitstream to determine first syntax flag information; if the first syntax flag information indicates that a preset intra prediction mode is to be applied to a current block, construct a block vector list for the current block; and determine a prediction block for the current block according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample precision block vector, where the subsample precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0012] According to a seventh aspect, a decoder is provided in embodiments of the disclosure. The decoder includes a second memory and a second processor. The second memory is configured to store computer programs executable by the second processor. The second processor is configured to, when executing the computer programs, perform the method described in the first aspect.

[0013] According to an eighth aspect, a computer-readable storage medium is provided in embodiments of the disclosure. The computer-readable storage medium is

configured to store computer programs. The computer programs, when executed by at least one processor, are operable to implement the method described in the first aspect or the method described in the second aspect.

[0014] Embodiments of the disclosure provide a coding method, a bitstream, an encoder, a decoder, and a storage medium. If the preset intra prediction mode is to be applied to the current block, the block vector list for the current block is constructed; and the prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes the block vector for the reference block corresponding to the guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in the preset template-error calculation method, and the preset template-error calculation method is determined according to the error calculation method for the template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream. That is, in embodiments of the disclosure, the construction and/or usage for the block vector list during intra prediction are adjusted and improved. By constructing AR-BVP candidate block vectors multiple times, and/or directly using a block vector for a reference block as a candidate block vector to be added to the block vector list, and/or using a subsample-precision block vector list for intra prediction, and/or reordering the block vector list in different template-error calculation methods, diversity of the block vector list can be improved, thereby improving intra prediction efficiency and improving coding performance.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] FIG. 1 is a block diagram illustrating a hybrid encoding framework.

[0016] FIG. 2 is a schematic diagram illustrating neighbouring blocks and non-neighbouring blocks.

[0017] FIG. 3 is schematic diagram I illustrating implementation of auto-relocated block vector prediction (AR-BVP) technology.

[0018] FIG. 4 is schematic diagram II illustrating implementation of AR-BVP technology.

[0019] FIG. 5 is schematic diagram I illustrating implementation of Intra template matching prediction (IntraTMP) technology.

[0020] FIG. 6 is schematic diagram II illustrating implementation of IntraTMP technology.

[0021] FIG. 7 is a schematic diagram illustrating implementation of spatial geometric partitioning mode (SGPM) technology.

[0022] FIG. 8 is a schematic diagram illustrating implementation of decoder-side intra mode derivation (DIMD) technology.

[0023] FIG. 9 is a schematic diagram illustrating implementation of template-based intra mode derivation (TIMD) technology.

[0024] FIG. 10 is a schematic block diagram of an encoder provided in embodiments of the disclosure.

[0025] FIG. 11 is a schematic block diagram of a decoder provided in embodiments of the disclosure.

[0026] FIG. 12 is a schematic network architectural diagram of a coding system provided in embodiments of the disclosure.

[0027] FIG. 13 is a schematic flowchart of a decoding method provided in embodiments of the disclosure.

[0028] FIG. 14 is a schematic diagram illustrating a block vector for a reference block provided in embodiments of the disclosure.

[0029] FIG. 15 is a schematic diagram illustrating templates provided in embodiments of the disclosure.

[0030] FIG. 16 is a schematic flowchart of an encoding method provided in embodiments of the disclosure.

[0031] FIG. 17 is a schematic structural diagram of an encoder provided in embodiments of the disclosure.

[0032] FIG. 18 is a schematic hardware structural diagram of an encoder provided in embodiments of the disclosure.

[0033] FIG. 19 is a schematic structural diagram of a decoder provided in embodiments of the disclosure.

[0034] FIG. 20 is a schematic hardware structural diagram of a decoder provided in embodiments of the disclosure.

[0035] FIG. 21 is a schematic structural diagram of a coding system provided in embodiments of the disclosure.

DETAILED DESCRIPTION

[0036] To facilitate more comprehensive understanding of the features and technical content of embodiments of the disclosure, a detailed explanation of the implementation of embodiments of the disclosure is given below in conjunction with accompanying drawings. The accompanying drawings are for illustration only rather than limiting embodiments of the disclosure.

[0037] Unless otherwise defined, all technical and scientific terms used in the disclosure have the same meanings as those commonly understood by those skilled in the technical field that the disclosure relates to. The terminology used in the disclosure is intended only for describing embodiments of the disclosure rather than limiting the disclosure.

[0038] In the following elaboration, “some embodiments” describes a subset of all possible embodiments. However, it should be understood that “some embodiments” can be the same subset or different subsets of all possible embodiments and can be combined with each other without conflict.

[0039] It should also be pointed out that terms “first/second/third” used in embodiments of the disclosure are merely for distinguishing similar objects and do not represent a specific

order of the objects. It can be understood that the specific order or sequence of "first/second/third" is interchangeable when allowed, so that embodiments of the disclosure described herein can be implemented in an order other than that illustrated or described herein.

[0040] In a video picture, a first colour component, a second colour component, and a third colour component are generally used to represent a coding block (CB). The three colour components are a luma component, a blue chroma component, and a red chroma component. Specifically, the luma component is usually denoted by symbol Y, the blue chroma component is usually denoted by symbol Cb or U, and the red chroma component is usually denoted by symbol Cr or V. Thus, a video picture can be represented in YCbCr format or YUV format.

[0041] In video coding standards, a block-based hybrid encoding framework can be adopted. Specifically, a picture in a video is partitioned into square largest coding units (LCUs) or coding tree units (CTUs) of the same size (such as 128×128, 64×64, etc.). Each LCU or CTU can be partitioned into rectangular CUs according to rules. A CU may be partitioned into prediction units (PUs), transform units (TUs), and the like.

[0042] Specifically, as illustrated in FIG. 1, the hybrid encoding framework includes a prediction module 11, a transform and quantization module 12, an entropy coding module 13, an inverse quantization and inverse transform module 14, a loop filtering module 15, and a decoded picture buffer module 16. The prediction module 11 can include an intra prediction module 11a and an inter prediction module 11b, where the inter prediction module 11b can include a motion estimation module and a motion compensation module. Since there is strong correlation between neighbouring samples in a picture of a video, in video coding technology, spatial redundancy between neighbouring samples is removed by applying an intra prediction method. In addition, since there is strong similarity between neighbouring pictures in a video, in video coding technology, temporal redundancy between neighbouring pictures is removed by applying an inter prediction method, thereby improving encoding efficiency.

[0043] Further, through transform, a predicted picture block is converted to a frequency domain, energy is redistributed. Through quantization, information insensitive to human

eyes is removed, thereby eliminating visual redundancy. Through entropy coding, character redundancy is eliminated based on a current context model and probability information in a binary bitstream.

[0044] The basic flow of a video coder is as follows. At an encoding end, after reading a black-and-white picture or a colour picture, the picture is partitioned into blocks, intra prediction or inter prediction is applied to a current block to generate a prediction block for the current block, the prediction block is subtracted from an original block of the current block to obtain a residual block, transform and quantization is performed on the residual block to obtain a quantized coefficient matrix, and entropy coding is performed on the quantized coefficient matrix and the entropy-coded quantized coefficient matrix is output into a bitstream. At a decoding end, intra prediction or inter prediction is applied to the current block to generate a prediction block for the current block. In addition, the bitstream is parsed to obtain the quantized coefficient matrix, inverse quantization and inverse transform are performed on the quantized coefficient matrix to obtain a residual block, and addition is performed on the prediction block and the residual block to obtain a reconstructed block. The reconstructed blocks form a reconstructed picture, and loop filtering is performed on the reconstructed picture on a picture basis or block basis to obtain a decoded picture.

[0045] At the encoding end, operations similar to those at the decoding end need to be performed to obtain the decoded picture. The decoded picture can serve as a reference picture for inter prediction of subsequent pictures.

[0046] Block partitioning information and mode information or parameter information such as prediction, transform, quantization, entropy coding, and loop filtering determined by the encoding end, if necessary, need to be output into the bitstream. At the decoding end, the same block partitioning information and mode information or parameter information such as prediction, transform, quantization, entropy coding, and loop filtering as those at the encoding end is determined by parsing the bitstream and analyzing according to existing information, thereby ensuring that the decoded picture obtained at the encoding end is the same as the decoded picture obtained at the decoding end.

[0047] During prediction, the current block can be partitioned into prediction units, and during transform, the current block can be partitioned into transform units, where partitioning

into prediction units can be different from partitioning into transform units. The above is the basic flow of a video coder under a block-based hybrid encoding framework. With development of technology, some modules or steps of this framework or flow may be optimized. Embodiments of the disclosure are applicable to the basic flow of a video coder under the block-based hybrid encoding framework, but are not limited to such framework and flow.

[0048] In addition, in embodiments of the disclosure, a current block (CB) can be a current coding unit, a current prediction unit, or a current transform unit, etc. Due to the need for parallel processing, a picture can be partitioned into slices, etc., and slices in the same picture can be processed in parallel, that is, there is no data dependency between the slices. “Frame” is a commonly used term, and it can generally be understood that one frame is one picture. The frame described in embodiments of the disclosure can also be replaced by “picture” or “slice”, etc.

[0049] Several prediction technologies in the related art are described in detail below.

[0050] (1) Intra block copy (IBC)

[0051] IBC technology is an intra prediction technology in which prediction samples are obtained based on a block vector. Similar to inter prediction, prediction is implemented based on a block vector directed to a reference block from a current block. The difference lies in that a reference block in inter prediction comes from a reconstructed picture that has been encoded, while a reference block in IBC comes from a reconstructed part of a current picture. Block vector information needs to be transmitted via a bitstream. Therefore, similar to intra prediction, there are an IBC-advanced motion vector prediction (IBC-AMVP) mode and an IBC-Merge mode.

[0052] IBC-AMVP mode: a reference block for the current block and a corresponding block vector (BV) are obtained through searching at an encoding end, and block vector prediction (BVP) is derived by constructing a candidate list (AMVP list). An index value *BVP index* of the BVP in the list and a difference BVD between the BVP and the BV are signalled into a bitstream, so that the BV can be reconstructed at a decoding end to complete prediction for the current block.

[0053] IBC-Merge mode: prediction is performed based on a constructed merge candidate list. The best merge candidate is selected from the list through encoding processes such as sum of absolute transformed difference (SATD) and rate distortion optimization (RDO), and a block vector for the best merge candidate is directly used to obtain a reference block. Only an index *merge index* of the merge candidate in the list needs to be encoded, and a BV can be obtained at a decoding end to complete prediction for the current block.

[0054] Candidates in the AMVP/Merge list can be obtained through the following methods:

1. a block vector inherited from a neighbouring block;

2. a block vector inherited from a non-neighbouring block;

3. a block vector inherited from a history-based block vector buffer list;

4. auto-relocated block vector prediction (AR-BVP) constructed based on existing block vectors;

5. a block vector obtained by averaging existing block vectors;

6. a default block vector (related to a block size).

[0055] FIG. 2 is a schematic diagram illustrating neighbouring blocks and non-neighbouring blocks. As illustrated in FIG. 2, a dot-filled small block is a current block, positions marked 1 to 5 are referred to as neighbouring blocks, and the rest are referred to as non-neighbouring blocks.

[0056] For AR-BVP technology, a guiding block vector (guiding BV) $\mathrm{BV}_{0,1}$ can be determined for a current coding block B_0, and a reference block B_1 can be determined according to the block vector. If there is a block vector $\mathrm{BV}_{1,2}$ in a region of the reference block B_1, a new block vector $\mathrm{BV}_{0,2} = \mathrm{BV}_{0,1} + \mathrm{BV}_{1,2}$ can be constructed as a candidate block vector for the current coding block. By analogy, a reference block B_n can be determined according to $\mathrm{BV}_{0,n}$, and if there is a block vector $\mathrm{BV}_{n,n+1}$ in a region of B_n, a new block vector $\mathrm{BV}_{0,n+1} = \mathrm{BV}_{0,1} + \mathrm{BV}_{1,2} + \cdots + \mathrm{BV}_{n,n+1}$ can be constructed, i.e., it can be considered that a new AR-BVP is constructed based on a constructed AR-BVP.

[0057] FIG. 3 is schematic diagram I illustrating implementation of AR-BVP technology. As illustrated in FIG. 3, a guiding block vector for a current coding block B_0 is $BV_{0,1}$, and a reference block B_1 is determined according to the block vector. If there is a block vector $BV_{1,2}$ in a region of the reference block B_1, a new block vector $BV_{0,2} = BV_{0,1} + BV_{1,2}$ can be constructed as a candidate block vector for the current coding block. A reference block B_2 can be determined according to $BV_{0,2}$, and if there is a block vector $BV_{2,3}$ in a region of B_2, a new block vector $BV_{0,3} = BV_{0,1} + BV_{1,2} + BV_{2,3}$ can be constructed. A reference block B_3 can be determined according to $BV_{0,3}$.

[0058] FIG. 4 is schematic diagram II illustrating implementation of AR-BVP technology. As illustrated in FIG. 4, when determining a block vector $BV_{n,n+1}$ corresponding to a reference block $BV_{n,n+1}$, multiple positions can be checked. For example, the center, top-left, top-right, bottom-left, and bottom-right of the reference block are checked in sequence to determine whether there is a coding block predicted by applying an IBC mode or intra template matching prediction (IntraTMP) mode, and if there is such block, a block vector stored for the block is taken as the block vector $BV_{n,n+1}$.

[0059] After the AMVP/Merge list is constructed, the list can be reordered according to template error values for candidates in the list. The template error value can be obtained according to a template error value, for example, a sum of absolute difference (SAD) value, between a reference block template for a candidate and a current block template.

[0060] (2) IntraTMP

[0061] IntraTMP technology is a special intra prediction coding tool. IntraTMP is implemented mainly through the following procedure: an L-shaped portion of neighbouring reconstructed samples for a current block is selected as a template, the most similar template is searched for within a given reconstructed region of a current picture, and a reconstructed block corresponding to the most similar template is used as a matching block and then used as a prediction block for the current block.

[0062] FIG. 5 is schematic diagram I illustrating implementation of IntraTMP technology. As illustrated in FIG. 5, R1 to R4 in the figure are available IntraTMP search regions. A

matching block is searched for in a sample-by-sample manner in these regions sequentially through raster scan.

[0063] Here, a coder indicates, via a flag intra_tmp_flag, whether to encode the current block by applying an IntraTMP mode. If yes, the same template matching process is performed at a decoding end to obtain the same prediction block at the decoding end, and there is no need to encode block vector information directed to the matching block from the current block.

[0064] FIG. 6 is schematic diagram II illustrating implementation of IntraTMP technology. As illustrated in FIG. 6, one search method for IntraTMP is to divide a search process into two steps: sparse search first, and then refinement search. In sparse search, template matching is performed at a step size S (i.e., every S samples horizontally and vertically, where $S>1$). For example, instead of searching for a matching block in a sample-by-sample manner in the search region, subsample search is performed horizontally and vertically. For example, if a current block vector for template matching is $(X0, Y0)$, then a next block vector for template matching in the horizontal direction is $(X0+S, Y0)$, and the vertical coordinate of the one after next block vector for template matching is $Y0+S$. In refinement search, template matching is performed at a smaller step size S' in a neighbouring region of a best-matching block vector according to a search result of the sparse search.

[0065] IntraTMP can be combined with a merge list (IntraTMP with Merge Candidates), and the search process in IntraTMP or an IntraTMP candidate block vector list can be extended based on block vector information in the merge list. For example, a sparse list is obtained through sparse search, and a merge list is constructed and merged with the sparse list to obtain a larger candidate block vector list. Refinement search is performed based on the new list to improve search effect.

[0066] (3) Spatial geometric partitioning mode (SGPM)

[0067] SGPM is an intra mode similar to geometric partitioning mode (GPM), in which weighted fusion can be performed on two intra prediction blocks, where weight values can correspond to different GPMs.

[0068] FIG. 7 is a schematic diagram illustrating implementation of SGPM technology. As illustrated in FIG. 7, in an implementation of SGPM, one GPM (partitioning mode) is combined with two intra modes (intra mode0 and intra mode1) as a prediction candidate for SGPM. An intra mode list can be constructed according to an intra mode(s) for a neighbouring block(s) and an intra mode(s) for a non-neighbouring block(s), and possible intra modes are obtained according to a constructed block vector list. These possible intra modes are combined with possible GPMs to construct a prediction candidate list for SGPM. The candidate list can be reordered according to template errors. When a flag for SGPM is true, a prediction candidate selected for the current block is determined according to an explicitly transmitted index value sgpm_index, thereby performing geometric partitioning prediction for the intra mode.

[0069] (4) Decoder-side intra mode derivation (DIMD)

[0070] A DIMD mode is a method for deriving an intra mode based on a histogram of gradient (HoG). In this method, gradient values are calculated in a template region for a current block and converted into angles, thereby constructing a histogram of intra angular modes. Several intra angular modes with highest magnitudes can be obtained according to the histogram.

[0071] FIG. 8 is a schematic diagram illustrating implementation of DIMD technology. As illustrated in FIG. 8, in a common DIMD implementation, for a DIMD coding block, multiple intra angular modes are derived from HoG, and in addition, an optimal non-angular mode is derived by constructing a block vector list and calculating template errors. Weighted fusion is performed on prediction blocks in these modes to obtain a final prediction block.

[0072] (5) Template-based intra mode derivation (TIMD) technology

[0073] TIMD technology is a method for intra mode derivation based on an error value for an intra mode in a template region. In this method, a most probable mode (MPM) list is constructed according to existing coding information, and prediction is performed on the template region by applying modes in the list, to obtain corresponding template error values, thereby deriving an optimal intra mode.

[0074] FIG. 9 is a schematic diagram illustrating implementation of TIMD technology. As illustrated in FIG. 9, similar to a DIMD method, during implementation of TIMD, two prediction modes with the smallest template error values and one intra non-angular mode can be used to generate prediction blocks, and weighted fusion is performed on these prediction blocks to obtain a final prediction block.

[0075] With most of common intra prediction methods, block-vector-based prediction can be derived by constructing a block vector list. Therefore, construction and usage of a block vector list have significant impact on intra prediction efficiency. In the related art, there are some problems, for example, AR-BVP candidate construction is relatively fixed when constructing a block vector list; only integer-sample block vectors are taken into consideration when using block vectors in methods such as SGPM and DIMD; and a reordering process for a block vector list is relatively fixed, etc.

[0076] That is, current approaches for constructing a block vector list are relatively fixed and are poor in flexibility, and thus are poorly adaptable to different prediction scenarios, which degrades intra prediction efficiency and thus affects coding performance.

[0077] Based on this, embodiments of the disclosure provide a coding method, a bitstream, an encoder, a decoder, and a storage medium. If a preset intra prediction mode is to be applied to a current block, a block vector list for the current block is constructed; and a prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the

current block or second syntax flag information transmitted in a bitstream. That is, in embodiments of the disclosure, the construction and/or usage process for the block vector list during intra prediction can be adjusted and improved. By constructing AR-BVP candidate block vectors multiple times, and/or directly using a block vector for a reference block as a candidate block vector to be added to the block vector list, and/or using a subsample-precision block vector list for intra prediction, and/or reordering the block vector list in different template-error calculation methods, diversity of the block vector list can be improved, thereby improving intra prediction efficiency and improving coding performance.

[0078] Embodiments of the disclosure will be described in detail below in conjunction with the accompanying drawings.

[0079] FIG. 10 is a schematic block diagram of an encoder provided in embodiments of the disclosure. As illustrated in FIG. 10, the encoder 100 can include a transform and quantization unit 101, an intra estimation unit 102, an intra prediction unit 103, a motion compensation unit 104, a motion estimation unit 105, an inverse transform and inverse quantization unit 106, a filter control analysis unit 107, a filtering unit 108, a coding unit 109, and a decoded picture buffer unit 110. The filtering unit 108 can implement deblocking filtering and sample adaptive offset (SAO) filtering. The coding unit 109 can implement header information encoding and context-based adaptive binary arithmetic coding (CABAC). For an input original video signal, one video coding block can be obtained through partitioning of a coding tree unit (CTU). Then, for residual sample information obtained after intra prediction or inter prediction, the video coding block is transformed by the transform and quantization unit 101, including transforming the residual information from the sample domain to the transform domain, and the obtained transform coefficients are quantized, so as to further reduce bit rate. The intra estimation unit 102 and the intra prediction unit 103 are used to perform intra prediction on the video coding block. Specifically, the intra estimation unit 102 and the intra prediction unit 103 are used to determine an intra prediction mode to-be-used to encode the video coding block. The motion compensation unit 104 and the motion estimation unit 105 are used to perform inter prediction encoding on the received video coding block relative to one or more blocks in one or more reference pictures, to provide temporal prediction information. Motion estimation performed by the motion

estimation unit 105 is a process of generating a motion vector, where the motion vector can be used to estimate motion of the video coding block. Then, the motion compensation unit 104 performs motion compensation based on the motion vector determined by the motion estimation unit 105. After the intra prediction mode is determined, the intra prediction unit 103 is further used to provide selected intra prediction data to the coding unit 109, and the motion estimation unit 105 sends calculated and determined motion vector data to the coding unit 109. In addition, the inverse transform and inverse quantization unit 106 is used for reconstruction of the video coding block. A residual block is reconstructed in the sample domain, and blocking artifacts of the reconstructed residual block are removed by the filter control analysis unit 107 and the filtering unit 108. Then, the reconstructed residual block is added to a prediction block in a picture of the decoded picture buffer unit 110, to generate a reconstructed video coding block. The coding unit 109 is used to encode various coding parameters and quantized transform coefficients. In the CABAC-based encoding algorithm, the context can be based on neighbouring coding blocks, and the coding unit 109 can be used to encode information indicating determined intra prediction mode and output a bitstream of the video signal. The decoded picture buffer unit 110 is used to store reconstructed video coding blocks for prediction reference. As the video picture encoding progresses, new reconstructed video coding blocks will be continuously generated, and these reconstructed video coding blocks will be stored into the decoded picture buffer unit 110.

[0080] FIG. 11 is a schematic block diagram of a decoder provided in embodiments of the disclosure. As illustrated in FIG. 11, the decoder 200 includes a decoding unit 201, an inverse transform and inverse quantization unit 202, an intra prediction unit 203, a motion compensation unit 204, a filtering unit 205, and a decoded picture buffer unit 206. The decoding unit 201 can implement header information decoding and CABAC decoding. The filtering unit 205 can implement deblocking filtering and SAO filtering. After the input video signal is encoded (as illustrated in FIG. 14), the bitstream of the video signal is output. The bitstream is input into the decoder 200. First, decoded transform coefficients are obtained by the decoding unit 201. The decoded transform coefficients are processed by the inverse transform and inverse quantization unit 202, so as to generate a residual block in the sample domain. The intra prediction unit 203 can be used to generate prediction data for a current

video decoding block based on the determined intra prediction mode and data from a previously decoded block of the current frame or picture. The motion compensation unit 204 is used to determine prediction information for the video decoding block by parsing motion vectors and other associated syntax elements, and use the prediction information to generate a prediction block for the video decoding block that is being decoded. A decoded video block is formed by summing the residual block from the inverse transform and inverse quantization unit 202 and the corresponding prediction block generated by the intra prediction unit 203 or the motion compensation unit 204. Blocking artifacts of the decoded video signal are removed by the filtering unit 205, which can improve quality of the video. The decoded video block is then stored into the decoded picture buffer unit 206. The decoded picture buffer unit 206 is used to store reference pictures used for subsequent intra prediction or motion compensation, and is also used to output the video signal, that is, the restored original video signal is obtained.

[0081] Further, FIG. 12 is a schematic network architectural diagram illustrating a coding system provided in embodiments of the disclosure. As illustrated in FIG. 12, the network architecture includes one or more electronic devices 31 to 3*N* and a communication network 01. The electronic devices 31 to 3*N* can perform video interaction with each other via the communication network 01. During implementation, the electronic devices can be various types of devices with video coding functions. For example, the electronic devices can include smartphones, tablet computers, personal computers, personal digital assistants (PDAs), navigation devices, digital telephones, video telephones, televisions, sensing devices, servers, etc., which is not specifically limited in embodiments of the disclosure.

[0082] In embodiments of the disclosure, a network architecture of a video coding system in which a decoding method and an encoding method are applied is provided herein. The decoder or the encoder in embodiments of the disclosure can be the above-mentioned electronic device. That is, the electronic device in embodiments of the disclosure has video coding functions, and can generally include a video encoder (i.e., encoder) and a video decoder (i.e., decoder).

[0083] It should also be noted that, embodiments of the disclosure are mainly applied to the intra prediction part illustrated in FIG. 10 and the intra prediction part illustrated in FIG.

11. That is, embodiments of the disclosure can be applied to an encoder, or can be applied to a decoder, or can even be applied to both an encoder and a decoder.

[0084] It should also be noted that, when embodiments of the disclosure are applied to the encoder illustrated in FIG. 10, the "current block" specifically refers to a coding block on which prediction is currently to be performed. When embodiments of the disclosure are applied to the decoder illustrated in FIG. 11, the "current block" specifically refers to a decoding block on which prediction is currently to be performed.

[0085] In an embodiment of the disclosure, FIG. 13 is a schematic flowchart of a decoding method provided in embodiments of the disclosure. As illustrated in FIG. 13, the method can include the following.

[0086] Step 1301, a bitstream is parsed to determine first syntax flag information.

[0087] It should be noted that, in embodiments of the disclosure, the method is applied to a decoder. Specifically, based on the structure of the decoder 200 illustrated in FIG. 15, the decoding method of embodiments of the disclosure can be applied to an intra prediction part, and is mainly applied to an intra prediction method based on block vector list construction, which can improve prediction efficiency of intra prediction.

[0088] It should also be noted that, in embodiments of the disclosure, the first syntax flag information indicates whether a preset intra prediction mode is to be applied to a current block. In some embodiments, if a value of the first syntax flag information is a first value, it is determined that the first syntax flag information indicates to apply the preset intra prediction mode to the current block; if the value of the first syntax flag information is a second value, it is determined that the first syntax flag information indicates not to apply the preset intra prediction mode to the current block.

[0089] It should also be noted that, in embodiments of the disclosure, the preset intra prediction mode can include at least one of: IBC, IntraTMP, SGPM, DIMD, or TIMD.

[0090] It can be understood that, in embodiments of the disclosure, the first syntax flag information can be represented differently with regard to different preset intra prediction modes. For example, for IBC, the corresponding first syntax flag information can be represented as cu_pred_mode; for DIMD, the corresponding first syntax flag information can

be represented as cu_dimd_flag; for IntraTMP, the corresponding first syntax flag information can be represented as cu_tmp_flag.

[0091] Here, the first value is different from the second value, and the first value and the second value each can be a parameter or a numerical form. Specifically, the first syntax flag information can be a parameter written in a profile, or can be a value of a flag, which is not specifically limited herein. Exemplarily, the first value can be 1 and the second value can be 0; alternatively, the first value can be 0 and the second value can be 1; alternatively, the first value can be true and the second value can be false; alternatively, the first value can be false and the second value can be true.

[0092] In an embodiment, taking the first value being 1 and the second value being 0 as an example, if a parsed value of cu_pred_mode is 1, it can be determined that the first syntax flag information indicates to apply the preset intra prediction mode to the current block; if the parsed value of cu_pred_mode is 0, it can be determined that the first syntax flag information indicates not to apply the preset intra prediction mode to the current block.

[0093] It should be noted that, in embodiments of the disclosure, the preset intra prediction mode can be a mode in which intra prediction is performed by constructing a block vector list.

[0094] In some embodiments, the preset intra prediction mode can include at least one of: IBC, IntraTMP, SGPM, DIMD, or TIMD.

[0095] In IBC, prediction is implemented based on a block vector directed to a reference block from the current block. In IntraTMP, a reconstructed block corresponding to the most similar template is determined as a matching block through template matching, which is then used as a prediction block for a current coding block. In SGPM, weighted fusion can be performed on two intra prediction blocks, where weight values can correspond to different GPMs. In DIMD, multiple intra angular modes are derived from HoG, and an optimal non-angular mode is derived by constructing a block vector list and calculating template errors, and weighted fusion is performed on prediction blocks generated in these modes to obtain a final prediction block. In TIMD, an MPM list is constructed according to existing coding information, and prediction is performed on a template region by applying

modes in the list, to obtain corresponding template error values, thereby deriving an optimal intra mode.

[0096] Step 1302, when the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block, a block vector list for the current block is constructed. The block vector list for the current block can satisfy at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0097] It should be noted that, in embodiments of the disclosure, when the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block, the block vector list for the current block can be constructed.

[0098] Further, in embodiments of the disclosure, the block vector list for the current block can satisfy at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes the block vector for the reference block corresponding to the guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting the template error values corresponding to the block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to the error calculation method

for the template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream.

[0099] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. When constructing the block vector list for the current block, different candidate block vectors can be obtained through at least two AR-BVP processes, and the obtained candidate block vectors are checked and then added to the block vector list. Between at least two AR-BVP processes, checking can be performed on at least one non-AR-BVP block vector.

[0100] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can include the block vector for the reference block corresponding to the guiding block vector for the current block. When constructing the block vector list for the current block, the block vector for the reference block determined based on the guiding block vector during an AR-BVP process can be directly used as a candidate block vector, and the candidate block vector is checked and then added to the block vector list.

[0101] It should be noted that, in embodiments of the disclosure, when the preset intra prediction mode is DIMD, or TIMD, or SGPM, the constructed block vector list for the current block can include a subsample-precision block vector. A block vector added to the block vector list for the current block can be of subsample precision, so that a prediction block of subsample precision can be obtained.

[0102] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be obtained by sorting the template error values corresponding to the block vectors in the block vector list. The preset template-error calculation method for calculating the template error values for the block vectors in the block vector list can be determined. The preset template-error calculation method can be determined according to the error calculation method for the template search process corresponding to the current block, or can be determined according to the second syntax flag information transmitted in the bitstream.

[0103] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector(s) in the block vector list for the current block can be determined as a first guiding block vector for the current block, and a first candidate block vector for the current block can be determined according to the first guiding block vector; where the first block vector includes at least one of: a block vector(s) for a neighbouring block(s) of the current block, a block vector(s) for a non-neighbouring block(s) of the current block, or a block vector(s) in a history-based motion vector prediction (HMVP) buffer. Then, the first candidate block vector and a second block vector(s) are added to the block vector list, where the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. Next, a third block vector in the block vector list is determined as a second guiding block vector for the current block, and a second candidate block vector for the current block is determined according to the second guiding block vector, where the third block vector at least includes the first candidate block vector. The second candidate block vector can be added to the block vector list.

[0104] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be constructed by using any number (that is, quantity) of block vectors among the block vector for the neighbouring block of the current block, the block vector for the non-neighbouring block of the current block, and the block vector in the HMVP buffer. The first block vector can be any one or more block vectors in the block vector list for the current block, i.e., the first block vector can be a non-AR-BVP block vector.

[0105] Accordingly, in embodiments of the disclosure, the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. The second block vector can be any number of block vectors among the block vector for the neighbouring block of the current block, the block vector for the non-neighbouring block of the current block, and the block vector in the HMVP buffer that are different from the first block vector.

[0106] It should be noted that, in embodiments of the disclosure, the second block vector can be a non-AR-BVP block vector that is determined after completing a first AR-BVP

process and then performing checking on at least one non-AR-BVP block vector, and that can be added to the block vector list.

[0107] Accordingly, in embodiments of the disclosure, the third block vector can include the first candidate block vector constructed through an AR-BVP process, and can further include the first block vector and/or the second block vector. That is, after the first candidate block vector and the second block vector are added to the block vector list for the current block, besides the first candidate block vector, other block vectors in the block vector list for the current block can also serve as guiding block vectors for the next AR-BVP process.

[0108] It can be understood that, in embodiments of the disclosure, taking constructing the block vector candidate list for the current block by performing an AR-BVP process twice as an example, during a first AR-BVP process, the first block vector in the block vector list for the current block can be used as the first guiding block vector, and then the corresponding first candidate block vector is determined according to the first guiding block vector, where the first candidate block vector is an AR-BVP block vector. After completing the first AR-BVP process, the obtained first candidate block vector (AR-BVP block vector) and the second block vector (a non-AR-BVP block vector different from the first block vector) can be added to the block vector list for the current block. Then, a second AR-BVP process is performed. The third block vector in the block vector list can be taken as the second guiding block vector, where the third block vector at least includes the first candidate block vector obtained from the first AR-BVP process. Then, the corresponding second candidate block vector can be determined according to the second guiding block vector, where the second candidate block vector is an AR-BVP block vector. Finally, the obtained second candidate block vector can be added to the block vector list.

[0109] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, more than two AR-BVP processes can also be implemented, which is not specifically limited in the disclosure. For example, after the obtained second candidate block vector is added to the block vector list, a guiding block vector for the next AR-BVP process can be determined according to the block vector list, and then a corresponding candidate block vector is determined according to the guiding block vector and added to the block vector candidate list.

[0110] Exemplarily, in some embodiments, it is assumed that when constructing the block vector list, AR-BVPs can be constructed twice. For example, candidate block vectors can be checked and added in the following order:

a) a block vector inherited from a neighbouring block;

b) a block vector inherited from a non-neighbouring block;

c) AR-BVP constructed based on existing block vectors;

d) a block vector inherited from a history-based block vector buffer list;

e) AR-BVP constructed based on existing block vectors;

f) a block vector obtained by averaging existing block vectors;

g) a default block vector.

[0111] The existing block vectors are some or all of the block vectors in the constructed block vector list. After adding the block vector for the neighbouring block/the block vector for the non-neighbouring block, perform AR-BVP for the first time. Thereafter, after adding the block vectors from the history-based block vector buffer list, perform AR-BVP for the second time.

[0112] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector in the block vector list for the current block can be determined as a first guiding block vector for the current block, and a first reference block corresponding to the first guiding block vector is determined, where the first block vector includes at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer. Then, a block vector for the first reference block is determined as a first candidate block vector for the current block, and/or the first guiding block vector and the block vector for the first reference block are summed to determine the first candidate block vector for the current block. Finally, the first candidate block vector can be added to the block vector list.

[0113] It should be noted that, in embodiments of the disclosure, when constructing the block vector list for the current block, a corresponding reference block can be determined according to a guiding block vector, and then a corresponding candidate block vector is

determined according to a block vector for the reference block. After completing an AR-BVP process, when determining the candidate block vector according to the block vector for the reference block, a result obtained by summing the guiding block vector and the block vector for the reference block can be determined as the candidate block vector and added to the block vector candidate list, or the block vector for the reference block can be directly determined as the candidate block vector and added to the block vector candidate list.

[0114] It can be understood that, in embodiments of the disclosure, when determining the first candidate block vector according to the first guiding block vector, the first reference block can be determined according to the first guiding block vector, and then the first candidate block vector is determined according to the block vector for the first reference block. The first candidate block vector can be determined by summing the first guiding block vector and the block vector for the first reference block, or the block vector for the first reference block can be directly determined as the first candidate block vector. Alternatively, the first candidate block vector can include both a result obtained by summing the first guiding block vector and the block vector for the first reference block, and the block vector for the first reference block.

[0115] That is, in embodiments of the disclosure, a block vector for a reference block determined according to a guiding block vector can be directly added to the block vector list as a candidate block vector, i.e., the block vector list for the current block includes the block vector for the reference block corresponding to the guiding block vector for the current block.

[0116] Exemplarily, in some embodiments, it is assumed that when constructing the block vector list, the block vector for the reference block can be directly used as a block vector candidate (candidate block vector). For a given guiding block vector BV_0 and a block vector BV_ref for a corresponding reference block, a candidate block vector BV_arbvp can be obtained by summing the guiding block vector and the block vector for the reference block, i.e. BV_arbvp = BV_0 + BV_ref, or the block vector BV_ref for the reference block can be directly used as a candidate block vector, i.e. BV_ext = BV_ref. Alternatively, when constructing AR-BVPs, after obtaining the vector for the reference block according to the guiding block vector each time, both BV_arbvp and BV_ext can be added to the block vector list as candidate block vectors.

[0117] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector in the block vector list for the current block can be determined as a first guiding block vector(s) for the current block, and a first reference block corresponding to the first guiding block vector is determined, where the first block vector includes at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer. Then, a block vector for the first reference block is determined as a first candidate block vector for the current block, and/or the first guiding block vector and the block vector for the first reference block are summed to determine the first candidate block vector for the current block. Next, the first candidate block vector and a second block vector are added to the block vector list, where the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. Then, a third block vector in the block vector list is determined as a second guiding block vector(s) for the current block, and a second reference block corresponding to the second guiding block vector is determined, where the third block vector at least includes the first candidate block vector. Next, a block vector for the second reference block is determined as a second candidate block vector for the current block, and/or the second guiding block vector and the block vector for the second reference block are summed to determine the second candidate block vector for the current block. Finally, the second candidate block vector can be added to the block vector list.

[0118] It can be understood that, in embodiments of the disclosure, taking constructing the block vector candidate list for the current block through two AR-BVP processes as an example, during a first AR-BVP process, the first block vector in the block vector list for the current block can be used as the first guiding block vector, and then the corresponding first candidate block vector is determined according to the first guiding block vector. The first candidate block vector is an AR-BVP block vector, and the first candidate block vector can include the block vector for the first reference block corresponding to the first candidate block vector, and/or a result obtained by summing the first guiding block vector and the block vector for the corresponding first reference block. After completing the first AR-BVP process, the obtained first candidate block vector (AR-BVP block vector) and the second block vector (a non-AR-BVP block vector different from the first block vector) can be added to the block

vector list for the current block. Then, a second AR-BVP process is performed. The third block vector in the block vector list can be determined as the second guiding block vector, where the third block vector at least includes the first candidate block vector obtained from the first AR-BVP process. Then, the corresponding second candidate block vector can be determined according to the second guiding block vector. The second candidate block vector is an AR-BVP block vector, and the second candidate block vector can include the block vector for the second reference block corresponding to the second candidate block vector, and/or a result obtained by summing the second guiding block vector and the block vector for the corresponding second reference block. Finally, the obtained second candidate block vector can be added to the block vector list.

[0119] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, more than two AR-BVP processes can also be implemented, which is not specifically limited in the disclosure. For example, after the obtained second candidate block vector is added to the block vector list, a guiding block vector for the next AR-BVP process can be determined according to the block vector list, and then a corresponding candidate block vector is determined according to the guiding block vector and added to the block vector candidate list.

[0120] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, a block vector for a reference block corresponding to a guiding vector can be added to the block vector list for the current block during any one or more AR-BVP processes.

[0121] That is, in embodiments of the disclosure, when constructing the block vector list, AR-BVP can be constructed at least twice. In addition, during an AR-BVP process, a block vector for a reference block can be directly used as a candidate block vector.

[0122] Further, in embodiments of the disclosure, when performing an AR-BVP process to obtain a candidate block vector, a guiding block vector can be determined according to existing block vectors in the block vector list. The first guiding block vector in a first AR-BVP process and the second guiding block vector in a second AR-BVP process can be partially the same or completely different.

[0123] That is, in embodiments of the disclosure, processes of different AR-BVP construction can be the same or different. For example, during the first AR-BVP construction, all existing block vectors in the block vector list can be used as guiding block vectors; during the second AR-BVP construction, all existing block vectors in the list and newly added AR-BVPs can be used as guiding block vectors.

[0124] Exemplarily, in some embodiments, the first guiding block vector in the first AR-BVP process can include some or all of the block vectors in the block vector list. The second guiding block vector in the second AR-BVP process can include a non-AR-BVP block vector in the block vector list that is different from the first guiding block vector, and can further include a newly constructed AR-BVP block vector (i.e., the first candidate block vector) added to the block vector list after the first AR-BVP process.

[0125] Further, in embodiments of the disclosure, an AR-BVP process(es) can be performed according to the guiding block vector for the current block to obtain at least one candidate block vector, so as to construct the block vector list for the current block, until a preset termination condition is satisfied, where the preset termination condition includes at least one of: the number of AR-BVP processes is greater than or equal to a preset threshold, the number of candidate block vectors determined through AR-BVP processes is greater than or equal to a first number threshold, or the number of block vectors in the block vector list is greater than or equal to a second number threshold.

[0126] It can be understood that, in embodiments of the disclosure, the preset threshold, the first number threshold, and the second number threshold each can be an integer greater than 0 set in advance, which is not specifically limited in the disclosure. For example, the preset threshold can be 3, the first number threshold can be 5, and the second number threshold can be 15.

[0127] That is, in embodiments of the disclosure, after a guiding block vector (e.g., the first guiding block vector or the second guiding block vector) is determined, one or more AR-BVP block vectors can be constructed. For example, a reference block B_1 is determined according to a guiding block vector $BV_{0,1}$, and then a new block vector $BV_{0,2} = BV_{0,1} + BV_{1,2}$ is constructed according to a block vector $BV_{1,2}$ existing in a region of the reference block

B_1. Then, AR-BVP process can be continued, e.g., a reference block B_n can be determined according to $BV_{0,n}$, and if there is a block vector $BV_{n,n+1}$ in a region of B_n, a new block vector $BV_{0,n+1} = BV_{0,1} + BV_{1,2} + \cdots + BV_{n,n+1}$ can be constructed.

[0128] It should be noted that, in embodiments of the disclosure, the termination condition (preset termination condition) for the AR-BVP construction process can include but is not limited to: the number of iterations of AR-BVP construction, the number of AR-BVPs already added to the block vector list, and the current number of block vectors in the block vector list.

[0129] It should be noted that, in embodiments of the disclosure, processes of AR-BVP construction can be the same or different. For example, during the first AR-BVP construction, construction may not be continued based on already added AR-BVPs (the first candidate block vectors); during the second AR-BVP construction, each guiding block vector can be used to construct multiple AR-BVPs, i.e., construction can be continued based on a newly added AR-BVP (the second candidate block vector).

[0130] Further, in embodiments of the disclosure, when determining a candidate block vector for the current block according to a block vector for a reference block, summation is performed on the guiding block vector and a block vector for at least one reference block respectively to determine the candidate block vector; and/or summation is performed on the guiding block vector and at least one block vector for the reference block respectively to determine the candidate block vector.

[0131] It should be noted that, in embodiments of the disclosure, one or more corresponding reference blocks can be determined according to a guiding block vector, and a block vector(s) for the one or more corresponding reference blocks can also be determined according to the guiding block vector.

[0132] Exemplarily, in some embodiments, when determining the first candidate block vector for the current block according to the block vector for the first reference block, summation is performed on the first guiding block vector and a block vector for at least one first reference block respectively to determine the first candidate block vector; and/or

summation is performed on the first guiding block vector and at least one block vector for the first reference block respectively to determine the first candidate block vector.

[0133] That is, in embodiments of the disclosure, for one guiding block vector, at least one corresponding reference block can be determined at at least one position in a corresponding reference region. For any one of the reference blocks, the reference block can correspond to at least one block vector.

[0134] Exemplarily, in some embodiments, FIG. 14 is a schematic diagram illustrating a block vector for a reference block provided in embodiments of the disclosure. As illustrated in FIG. 14, when determining a block vector corresponding to a reference block, multiple positions can be checked. For example, the center, top-left, top-right, bottom-left, and bottom-right of the reference block are checked in sequence to determine whether there is a coding block, and if there is such block, a block vector stored for the block is taken as the block vector. Any one reference block may correspond to one or more block vectors. For example, two block vectors are stored for a reference block at the center position.

[0135] Further, in embodiments of the disclosure, the number of block vectors in the HMVP buffer to be added to the block vector list can be determined according to the number of block vectors in the block vector list.

[0136] That is, in embodiments of the disclosure, when constructing the block vector list, the number of block vectors in the HMVP buffer to be added to the block vector list can be adjusted according to the number of block vectors already added to the block vector list.

[0137] Exemplarily, in some embodiments, assuming that a maximum length of the block vector list is N, and there are already n block vectors in the block vector list before adding block vectors in the HMVP buffer, then the number of block vectors in the HMVP buffer to be added to the block vector list can be set according to N and n. For example, the number of block vectors in the HMVP buffer to be added to the block vector list is determined to be $(N - n)/2$.

[0138] Further, in embodiments of the disclosure, before adding any block vector (an AR-BVP block vector or a non-AR-BVP block vector) to the block vector list, the block vector is checked. If the block vector does not satisfy a preset addition condition, the block vector

will not be added to the block vector list; if the block vector satisfies the preset addition condition, the block vector will be added to the block vector list. The preset addition condition includes at least one of: the block vector is different from any block vector in the block vector list; or a reference block corresponding to the block vector exists.

[0139] Exemplarily, in some embodiments, between at least two AR-BVP processes, at least one non-AR-BVP block vector is checked. For example, a block vector (non-AR-BVP block vector) in the HMVP buffer is checked to determine whether the block vector in the HMVP buffer can be added to the block vector list for the current block. If the block vector in the HMVP buffer is the same as a block vector already added to the list, the non-AR-BVP block vector will not be added, i.e., for a non-AR-BVP block vector checked, if the addition condition is not satisfied, the non-AR-BVP block vector will not be added.

[0140] Accordingly, in some embodiments, between at least two AR-BVP processes, after at least one non-AR-BVP block vector is checked, if a certain non-AR-BVP block vector is different from all block vectors already added to the list and a reference block corresponding to the non-AR-BVP block vector exists, it can be determined that the non-AR-BVP block vector can be added to the block vector list for the current block.

[0141] Step 1303, a prediction block for the current block is determined according to the block vector list for the current block.

[0142] In embodiments of the disclosure, when the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block, after the block vector list for the current block is constructed, the prediction block for the current block can be determined according to the block vector list for the current block.

[0143] Further, in embodiments of the disclosure, when the preset intra prediction mode is DIMD, or TIMD, or SGPM, the constructed block vector list for the current block can include a subsample-precision block vector. Accordingly, when determining the prediction block for the current block according to the block vector list for the current block, template error values for block vectors in the block vector list can be determined according to the subsample-precision block vector in the block vector list; and then the prediction block for

the current block can be determined according to the template error values for the block vectors in the block vector list.

[0144] It can be understood that, in embodiments of the disclosure, for a block vector list of subsample precision ("subsample-precision block vector list" for short), each block vector in the block vector list corresponds to a subsample precision parameter, which can characterize a precision value of the block vector. The subsample precision parameter can be a value greater than 0 and less than or equal to 1, for example, the subsample precision parameter for a block vector is 1/16.

[0145] Further, in embodiments of the disclosure, when determining the template error values for the block vectors in the block vector list according to the subsample-precision block vector in the block vector list, a template type parameter and a template size parameter corresponding to the template type parameter can be determined; then, an adjusted block vector corresponding to the subsample-precision block vector is determined according to the template type parameter, the template size parameter, and a subsample bit parameter for the subsample-precision block vector; prediction values for a reference template are determined according to the adjusted block vector; and the template error values for the block vectors in the block vector list can be determined according to the prediction values for the reference template and reconstructed values for a current template.

[0146] It should be noted that, in embodiments of the disclosure, the template type parameter can be used to determine the type of a template. According to the template type parameter, the type of a corresponding template can be determined to be a top template or a left template.

[0147] It should be noted that, in embodiments of the disclosure, the template size parameter can be used to determine the size of a template. According to the template size parameter, the width or height of a corresponding template can be determined. For a top template, the corresponding width is a width value of the current block, and the corresponding height can be determined according to the template size parameter; for a left template, the corresponding height is a height value of the current block, and the corresponding width can be determined according to the template size parameter.

[0148] Exemplarily, in some embodiments, FIG. 15 is a schematic diagram illustrating templates provided in embodiments of the disclosure. As illustrated in FIG. 15, the current block has a height value H and a width value W. Two template types, such as a top template and a left template, corresponding to the current block are determined according to the template type parameter. Based on the template size parameter for the top template, the width of the top template can be determined to be Tw; based on the template size parameter for the left template, the height of the left template can be determined to be Th.

[0149] It should be noted that, in embodiments of the disclosure, when determining the adjusted block vector corresponding to the subsample-precision block vector according to the template type parameter, the template size parameter, and the subsample bit parameter for the subsample-precision block vector, a template corresponding to the current block and a size value of the template can be determined according to the template type parameter and the template size parameter, and then adjustment is performed according to the subsample bit parameter corresponding to the block vector and the size value of the template, to obtain the adjusted block vector.

[0150] Exemplarily, in some embodiments, assuming that the template corresponding to the current block is determined to be a top template according to the template type parameter, the size value of the corresponding template is the height Th, and the subsample bit parameter for the corresponding block vector BV is mv_bits, then the block vector BV can be adjusted according to the following formula to obtain the corresponding adjusted block vector BV_top:

$$BV_top = BV + (Th << mv_bits) \quad (1)$$

[0151] Exemplarily, in some embodiments, assuming that the template corresponding to the current block is determined to be a left template according to the template type parameter, the size value of the corresponding template is the width Tw, and the subsample bit parameter for the corresponding block vector BV is mv_bits, then the block vector BV can be adjusted according to the following formula to obtain the corresponding adjusted block vector BV_left:

$$BV_left = BV + (Tw << mv_bits) \quad (2)$$

[0152] Further, in embodiments of the disclosure, when determining the prediction values for the reference template according to the adjusted block vector, if a candidate reference template corresponding to the adjusted block vector is available, the prediction values for the reference template can be determined according to the adjusted block vector and a subsample interpolation algorithm; if the candidate reference template corresponding to the adjusted block vector is unavailable, the prediction values for the reference template can be determined according to block vectors in the block vector list and the subsample interpolation algorithm.

[0153] It can be understood that, in embodiments of the disclosure, a corresponding candidate reference template can be determined based on the adjusted block vector. If the candidate reference template is available, the prediction values for the reference template can be directly determined according to the adjusted block vector and the subsample interpolation algorithm; if the candidate reference template is unavailable, the adjusted block vector is no longer used, and instead, the prediction values for the reference template are determined according to a block vector before adjustment and the subsample interpolation algorithm.

[0154] Exemplarily, in some embodiments, assuming that the adjusted block vector is BV_top, a candidate reference template with a width of W and a height of Th is determined according to BV_top. If the candidate reference template is available, the prediction values for the corresponding reference template can be calculated based on BV_top by means of the subsample interpolation algorithm; if the candidate reference template is unavailable, BV_top can be adjusted to be equal to BV, and the prediction values for the corresponding reference template can be calculated by means of the subsample interpolation algorithm.

[0155] It should be noted that, in embodiments of the disclosure, the subsample interpolation algorithm can be implemented with an interpolation filter, where the interpolation filter can be a 2-tap interpolation filter, which can be the same as a 2-tap interpolation filter used for inter prediction or IBC prediction.

[0156] Accordingly, in embodiments of the disclosure, prediction on a left template region can also be performed through the above method, i.e., the prediction values for the corresponding reference template is determined according to BV_left. As such, the template

error values corresponding to the block vectors can be calculated according to the prediction values for the reference template and the reconstructed values for the current template, and then the prediction block for the current block can be determined according to the template error values for the block vectors in the block vector list.

[0157] Exemplarily, in some embodiments, assuming that the preset intra prediction mode is DIMD, a template error value in a predefined intra non-angular mode can be calculated. For example, one mode Planar is predefined, and a template error value in the Planar mode is calculated according to reference samples. Then, an optimal non-angular prediction method can be selected according to the candidate block vectors in the block vector list and the template error value in the intra non-angular mode. If the optimal prediction method is to use a certain candidate block vector, in a case that the block vector is of subsample precision, a reference block for the current block can be obtained by means of interpolation filtering, where an interpolation filter can adopt an *n*-tap filter, for example, *n* = 8, which can be the same as an interpolation filter used for IBC prediction. After the optimal non-angular prediction method is determined, a prediction block generated in the optimal non-angular prediction method is fused with prediction blocks generated in intra angular modes derived from HoG, to obtain a final prediction block.

[0158] Further, in embodiments of the disclosure, the block vector list for the current block can be obtained by sorting template error values corresponding to block vectors in the block vector list. Accordingly, when determining the prediction block for the current block according to the block vector list for the current block, block-vector template-error calculation can be performed on block vectors in the block vector list to determine template error values for the block vectors in the block vector list; then, the block vector list can be sorted according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list; and the prediction block for the current block can be determined based on the sorted block vector list.

[0159] It should be noted that, in embodiments of the disclosure, a preset template-error calculation method used for calculating the template error values for the block vectors in the block vector list can be determined. The preset template-error calculation method can be determined according to an error calculation method for a template search process

corresponding to the current block, or can be determined according to the second syntax flag information transmitted in the bitstream.

[0160] It can be understood that, in embodiments of the disclosure, in an implementation, the second syntax flag information indicates that the preset template-error calculation method is SAD or mean-removed SAD (MR-SAD), which can be represented by cu_tmp_lic_flag. In some embodiments, if a value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SAD.

[0161] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is SATD or MR-SATD, which can be represented by cu_tmp_lic_flag. In some embodiments, if the value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SATD.

[0162] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is SATD or SAD, which can be represented by cu_tmp_satd_flag. In some embodiments, if the value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD.

[0163] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is one of SAD, MR-SAD, SATD, or MR-SATD, which can be represented jointly by cu_tmp_lic_flag and cu_tmp_satd_flag. In some embodiments, if a value of first

information cu_tmp_lic_flag in the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD or SATD; if the value of the first information cu_tmp_lic_flag in the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SAD or MR-SATD. If a value of second information cu_tmp_satd_flag in the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD or MR-SATD; if the value of the second information cu_tmp_satd_flag in the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD or MR-SAD. For example, if the value of cu_tmp_lic_flag is the first value and the value of cu_tmp_satd_flag is the first value, it can be determined that the preset template-error calculation method is SATD; if the value of cu_tmp_lic_flag is the second value and the value of cu_tmp_satd_flag is the first value, it can be determined that the preset template-error calculation method is MR-SATD.

[0164] Here, the first value is different from the second value, and the first value and the second value each can be a parameter or a numerical form. Specifically, the second syntax flag information can be a parameter written in a profile, or can be a value of a flag, which is not specifically limited herein. Exemplarily, the first value can be 1 and the second value can be 0; alternatively, the first value can be 0 and the second value can be 1; alternatively, the first value can be true and the second value can be false; alternatively, the first value can be false and the second value can be true.

[0165] In an embodiment, taking the first value being 1 and the second value being 0 as an example, if a parsed value of cu_tmp_lic_flag is 1, it can be determined that the second syntax flag information indicates that the preset template-error calculation method is SAD; if the parsed value of cu_tmp_lic_flag is 0, it can be determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SAD.

[0166] Further, in embodiments of the disclosure, when performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list, the bitstream can be

parsed to determine the second syntax flag information; then, a preset template-error calculation method is determined according to the second syntax flag information, where the preset template-error calculation method includes any one of: SAD, MR-SAD, SATD, or MR-SATD; and template error values for the block vectors in the candidate block vector list corresponding to at least one template type are determined based on the preset template-error calculation method.

[0167] It can be understood that, in embodiments of the disclosure, which one of SAD, MR-SAD, SATD, or MR-SATD is the preset template-error calculation method for performing block-vector template-error calculation can be determined according to the second syntax flag information.

[0168] Exemplarily, in some embodiments, assuming that the second syntax flag information is represented by cu_tmp_lic_flag, then according to a value of cu_tmp_lic_flag, whether the preset template-error calculation method is SAD or is MR-SAD can be determined, or whether the preset template-error calculation method is SATD or is MR-SATD can be determined.

[0169] Exemplarily, in some embodiments, assuming that the second syntax flag information is represented by cu_tmp_satd_flag, then according to a value of cu_tmp_satd_flag, whether the preset template-error calculation method is SAD or is SATD can be determined.

[0170] Exemplarily, in some embodiments, assuming that the second syntax flag information includes first information cu_tmp_lic_flag and second information cu_tmp_satd_flag, then which one of SAD, MR-SAD, SATD, or MR-SATD is the preset template-error calculation method can be determined according to a value of cu_tmp_lic_flag and a value of cu_tmp_satd_flag. For example, if the value of cu_tmp_lic_flag is a first value and the value of cu_tmp_satd_flag is a first value, it can be determined that the preset template-error calculation method is SATD; if the value of cu_tmp_lic_flag is a second value and the value of cu_tmp_satd_flag is a first value, it can be determined that the preset template-error calculation method is MR-SATD.

[0171] Further, in embodiments of the disclosure, when determining the preset template-error calculation method according to the second syntax flag information, a first template-error calculation method and a second template-error calculation method can be determined according to the second syntax flag information, where the first template-error calculation method is used for performing sparse search on the at least one template type, and the second template-error calculation method is used for performing refinement search on the at least one template type and block-vector template-error calculation.

[0172] That is, in embodiments of the disclosure, based on a value of the second syntax flag information (e.g., cu_tmp_lic_flag), the first template-error calculation method used for performing sparse search and the second template-error calculation method used for performing refinement search and block-vector template-error calculation can be determined, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage.

[0173] Exemplarily, in some embodiments, assuming that the value of the second syntax flag information cu_tmp_lic_flag is a first value, then it can be determined that MR-SAD (the first template-error calculation method) is to be used in a sparse search stage, MR-SATD (the second template-error calculation method) is to be used in a refinement search stage, and MR-SATD is to be used for block-vector template-error calculation.

[0174] Further, in embodiments of the disclosure, when determining the preset template-error calculation method according to the second syntax flag information, a third template-error calculation method and a fourth template-error calculation method can be determined according to the second syntax flag information, where the third template-error calculation method is used for performing sparse search on the at least one template type and block-vector template-error calculation, and the fourth template-error calculation method is used for performing refinement search on the at least one template type.

[0175] That is, in embodiments of the disclosure, the third template-error calculation method used for performing sparse search and block-vector template-error calculation and the fourth template-error calculation method used for performing refinement search can be determined based on a value of the second syntax flag information (e.g., cu_tmp_lic_flag),

i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a sparse search stage.

[0176] Exemplarily, in some embodiments, assuming that the value of the second syntax flag information cu_tmp_lic_flag is a second value, then it can be determined that SAD (the third template-error calculation method) is to be used in a sparse search stage, SATD (the fourth template-error calculation method) is to be used for block-vector template-error calculation, and SATD is to be used in a refinement search stage.

[0177] Further, in embodiments of the disclosure, when determining the preset template-error calculation method according to the second syntax flag information, a fifth template-error calculation method can be determined according to the second syntax flag information, where the fifth template-error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

[0178] That is, in embodiments of the disclosure, the fifth template-error calculation method used for performing sparse search, refinement search, and block-vector template-error calculation can be determined based on a value of the second syntax flag information (e.g., cu_tmp_lic_flag), i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage and a sparse search stage.

[0179] Exemplarily, in some embodiments, assuming that the value of the second syntax flag information cu_tmp_lic_flag is a first value, then it can be determined that MR-SAD (the fifth template-error calculation method) is to be used in a sparse search stage, MR-SAD is to be used in a refinement search stage, and MR-SAD is to be used for block-vector template-error calculation.

[0180] Further, in embodiments of the disclosure, the second syntax flag information includes first information and second information. When determining the preset template-error calculation method according to the second syntax flag information, a sixth error calculation method can be determined according to the first information and the second information, where the sixth error calculation method is used for performing sparse search

on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

[0181] That is, in embodiments of the disclosure, the sixth template-error calculation method used for performing sparse search, refinement search, and block-vector template-error calculation can be determined based on a value of the first information (e.g., cu_tmp_lic_flag) and a value of the second information (e.g., cu_tmp_satd_flag) in the second syntax flag information, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage and a sparse search stage, and the calculation method is determined according to the value of cu_tmp_lic_flag and the value of cu_tmp_satd_flag.

[0182] Exemplarily, in some embodiments, assuming that the value of the first information cu_tmp_lic_flag in the second syntax flag information is a first value, then it can be determined that SAD or SATD is to be used; if the value of cu_tmp_satd_flag is a first value, it can be further determined that SATD is to be used; if the value of cu_tmp_satd_flag is a second value, it can be further determined that SAD is to be used.

[0183] That is, in embodiments of the disclosure, both cu_tmp_satd_flag and cu_tmp_lic_flag flags can be encoded to determine which one of SAD, SATD, MR-SAD, or MR-SATD is to be used for search and template error calculation for a block vector list.

[0184] Further, when performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list, block-vector template-error calculation can be performed on the block vectors in the block vector list according to a preset template-error calculation method for sparse search in a template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list; alternatively, block-vector template-error calculation can be performed on the block vectors in the block vector list according to a preset template-error calculation method for refinement search in the template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list.

[0185] That is, in embodiments of the disclosure, the template-error calculation method for a sparse search stage can be directly used as the preset template-error calculation method for block-vector template-error calculation, or the template-error calculation method for a refinement search stage can be directly used as the preset template-error calculation method for block-vector template-error calculation.

[0186] Further, in embodiments of the disclosure, when sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list, the bitstream can be parsed to determine third syntax flag information; then, at least one template-number (that is, template-quantity) corresponding to the at least one template type is determined according to the third syntax flag information; next, the template error values for the block vectors in the candidate block vector list corresponding to the at least one template type are sorted separately according to the at least one template-number, to obtain at least one first candidate list corresponding to the at least one template type.

[0187] It should be noted that, in embodiments of the disclosure, for each of different template types, a template-number (that is, template-quantity) corresponding to the template type can be determined. Then, template error values for block vectors corresponding to the template type are sorted according to the template-number. As such, first candidate lists corresponding to different template types can be obtained, i.e., the sorted block vector list includes first candidate lists corresponding to different template types.

[0188] It can be understood that, in embodiments of the disclosure, the third syntax flag information indicates template-numbers corresponding to different template types, which can be represented by cu_tmp_idx. In some embodiments, according to a value of the third syntax flag information, a search result that needs to be maintained for each of different template types can be determined. For example, it is determined, according to the value of the third syntax flag information, that a search result corresponding to an L-shaped template, a search result corresponding to a top template, and a search result corresponding to a left template need to be maintained, i.e., a template-number corresponding to the L-shaped template is determined to be $M0$, a template-number corresponding to the top template is determined to be $M1$, and a template-number corresponding to the left template

is determined to be *M*2, where *M*0, *M*1, and *M*2 each are an integer greater than or equal to 0.

[0189] Exemplarily, in some embodiments, for the current block, MR-SAD is used to calculate template errors. When the value of the third syntax flag information cu_tmp_idx is greater than a threshold *N* (e.g., *N*=13), a search result corresponding to the L-shaped template, a search result corresponding to the top template, and a search result corresponding to the left template need to be maintained. Accordingly, MR-SAD is used to calculate template errors for the block vectors in the block vector list, and template error values diff0 corresponding to the L-shaped template, template error values diff1 corresponding to the top template, and template error values diff2 corresponding to the left template are obtained. According to diff0, diff1, and diff2 for the block vectors, the best *M*0 candidate block vectors corresponding to the L-shaped template, the best *M*1 candidate block vectors corresponding to the top template, and the best *M*2 candidate block vectors corresponding to the left template are obtained, which can be denoted as mrgList0, mrgList1, and mrgList2 respectively, where mrgList0 is a first candidate list corresponding to the L-shaped template, mrgList1 is a first candidate list corresponding to the top template, and mrgList2 is a first candidate list corresponding to the left template.

[0190] Further, in embodiments of the disclosure, when determining the prediction block for the current block based on the sorted block vector list, sparse search can be performed separately according to the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively; then, the at least one first candidate list is merged with the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type; next, refinement search can be performed separately according to the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; finally, a target candidate list corresponding to the current block can be determined according to the at least one fourth candidate list, and the prediction block for the current block can be determined according to the target candidate list.

[0191] Exemplarily, in some embodiments, sparse search is performed separately for the L-shaped template, the top template, and the left template, and search results are recorded as sparseList0, sparseList1, and sparseList2, where sparseList0 is a search result for template error values corresponding to the L-shaped template, sparseList1 is a search result for template error values corresponding to the top template, and sparseList2 is a search result for template error values corresponding to the left template. Here, sparseList0 is a second candidate list corresponding to the L-shaped template, sparseList1 is a second candidate list corresponding to the top template, and sparseList2 is a second candidate list corresponding to the left template.

[0192] Exemplarily, in some embodiments, the block vector list(s) and the sparse search result(s) are merged, i.e., the at least one first candidate list and the at least one second candidate list are merged respectively. For example, mrgList0 and sparseList0 are merged, i.e., the first *K* candidate block vectors with the smallest MR-SAD corresponding to the L-shaped template are retained, where *K* can be an integer greater than or equal to the maximum length of sparseList0, and the merged list is denoted as sparseList0'. Accordingly, mrgList1 and sparseList1 can be merged to obtain corresponding sparseList1', and mrgList2 and sparseList2 can be merged to obtain corresponding sparseList2'. Here, sparseList0' is a third candidate list corresponding to the L-shaped template, sparseList1' is a third candidate list corresponding to the top template, and sparseList2' is a third candidate list corresponding to the left template.

[0193] Exemplarily, in some embodiments, based on the third candidate lists obtained by merging, i.e., sparseList0' corresponding to the L-shaped template, sparseList1' corresponding to the top template, and sparseList2' corresponding to the left template, refinement search can be performed respectively to obtain a search result sparselist0' for template error values corresponding to the L-shaped template, a search result sparselist1' for template error values corresponding to the top template, and a search result sparselist2' for template error values corresponding to the left template, where sparselist0' is a fourth candidate list corresponding to the L-shaped template, sparselist1' is a fourth candidate list corresponding to the top template, and sparselist2' is a fourth candidate list corresponding to the left template.

[0194] Exemplarily, in some embodiments, when determining the target candidate list corresponding to the current block according to the at least one fourth candidate list, the at least one fourth candidate list can be merged, for example, sparselist0', sparselist1', and sparselist2' are merged to obtain a final target candidate list.

[0195] Further, in embodiments of the disclosure, after at least one second candidate list is obtained through sparse search, for any one of the second candidate lists, AR-BVP construction can be performed based on the second candidate list. A third guiding block vector can be determined according to the second candidate list, and then a third candidate block vector(s) for the current block is determined according to the third guiding block vector. Next, a template error value corresponding to the third candidate block vector is determined, and the second candidate list is updated according to the template error value corresponding to the third candidate block vector.

[0196] Exemplarily, in some embodiments, a block vector in a candidate list sparseList obtained through sparse search can be used as a guiding block vector (third guiding block vector) to construct AR-BVP candidates (third candidate block vectors), and then template error values for these AR-BVP candidates can be calculated and sparseList can be updated. AR-BVP candidate construction can be performed for sparseList corresponding to different template types, and sparseList is updated according to template error values for AR-BVP candidates.

[0197] Further, in embodiments of the disclosure, when determining the prediction block for the current block based on the sorted block vector list, sparse search can be performed separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively; then, a first candidate list corresponding to a preset template type is merged with the at least one second candidate list to obtain at least one third candidate list corresponding to the at least one template type; next, refinement search can be performed separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; finally, a target candidate list corresponding to the current block can be determined according to the at least

one fourth candidate list, and the prediction block for the current block can be determined according to the target candidate list.

[0198] It should be noted that, in embodiments of the disclosure, when merging a block vector list and sparse search result(s), the first candidate list corresponding to the preset template type can be merged with the at least one second candidate list, to obtain at least one third candidate list corresponding to the at least one template type. The preset template type can be any one of an L-shaped template, a top template, or a left template.

[0199] Exemplarily, in some embodiments, when merging the block vector list and the sparse search result(s), mrgList0 can be merged with each of sparseList0, sparseList1, and sparseList2, and the merged lists are denoted as sparseList0', sparseList1', and sparseList2'. sparseList0' is a third candidate list corresponding to the L-shaped template, sparseList1' is a third candidate list corresponding to the top template, and sparseList2' is a third candidate list corresponding to the left template.

[0200] That is, in embodiments of the disclosure, a block vector list mrgList can be constructed and reordered according to the preset template type (e.g., the L-shaped template), and a candidate list sparseList0 corresponding to the L-shaped template, a candidate list sparseList1 corresponding to the top template, and a candidate list sparseList2 corresponding to the left template can be obtained through sparse search. Subsequently, only mrgList is merged with each of sparseList0, sparseList1, and sparseList2.

[0201] Further, in embodiments of the disclosure, during refinement search based on at least one third candidate list, all block vectors in the third candidate list correspond to the same template search region. Alternatively, during refinement search based on at least one third candidate list, fourth block vectors in the third candidate list correspond to the same template search region, where the fourth block vectors are block vectors derived from the first candidate list.

[0202] It should be noted that, in embodiments of the disclosure, when performing refinement search separately for at least one template type based on at least one third candidate list, template search regions corresponding to block vectors in the third candidate list can be the same or different. For example, two template search regions corresponding

to any two block vectors in the third candidate list can be the same; or two template search regions corresponding to any two fourth block vectors in the third candidate list can be the same.

[0203] It can be understood that, in embodiments of the disclosure, the fourth block vectors can be block vectors in the third candidate list that are derived from the first candidate list. That is, when merging the first candidate list and the second candidate list, the fourth block vectors in the first candidate list constitute the third candidate list.

[0204] Further, in embodiments of the disclosure, when determining the prediction block for the current block according to the block vector list for the current block, the bitstream can be parsed to determine a block vector index identifier (ID) for the current block; then, a target block vector for the current block is determined according to the block vector index ID and the block vector list for the current block; finally, the prediction block for the current block can be determined according to the target block vector.

[0205] Exemplarily, in some embodiments, if the preset intra prediction mode is IBC, the prediction block for the current block can be determined according to the parsed block vector index ID in combination with the constructed block vector list for the current block. For example, in an IBC merge mode, a block vector is selected from the block vector list according to a parsed merge index (block vector index ID), and then a reference block for the current block is obtained as the prediction block.

[0206] Further, in embodiments of the disclosure, the bitstream is parsed to determine a residual block for the current block; then, a reconstructed block for the current block can be determined according to the residual block and the prediction block.

[0207] It can be understood that, in embodiments of the disclosure, after the residual block and the prediction block for the current block are determined, a sum of the residual block and the prediction block can be determined as the reconstructed block for the current block.

[0208] In summary, through the decoding method proposed in the above step 1301 to step 1303, the construction and/or usage process for the block vector list during intra prediction is adjusted and improved. When constructing the block vector list, AR-BVP

candidate block vectors can be constructed multiple times. In addition, a block vector for a reference block can also be directly used as a candidate block vector and added to the block vector list. Furthermore, a block vector list of subsample-precision can be used for intra prediction. Moreover, various methods can be applied to determine multiple template-error calculation methods, so the block vector list can be reordered by using different template-error calculation methods.

[0209] That is, the decoding method proposed in embodiments of the disclosure is a method for improving construction and/or usage of a block vector list, where the block vector list can be an AMVP/Merge list in an IBC mode, a Merge list in an IntraTMP mode, or a block vector list constructed to obtain block-vector-based prediction in other intra modes. During construction and/or use of the block vector list, diversity of the block vector list and utilization efficiency of the block vector list are improved by applying methods such as constructing AR-BVP candidate block vectors multiple times, and/or directly adding a block vector for a reference block to the block vector list, and/or using a block vector list of subsample-precision in intra modes, and/or sorting the block vector list by using different template-error calculation methods.

[0210] The decoding method proposed in embodiments of the disclosure is exemplarily described below by taking the IBC mode and the first syntax flag information being cu_pred_mode as an example. When constructing a block vector list, AR-BVPs can be constructed multiple times, and a block vector for a reference block can also be directly added to the block vector list as a block vector candidate.

[0211] Decoding process:

[0212] 1. A prediction mode cu_pred_mode for a current block is parsed. When the prediction mode is the IBC mode, a block vector list is constructed for a subsequent prediction process. For example, in a merge mode, a block vector list is constructed, and a block vector and prediction information (e.g., a local illumination compensation (LIC) flag, etc.) for the current block are determined according to a merge index. For example, in the AMVP mode, a block vector list is constructed, a predicted block vector for the current block is determined according to a BVP index, and a final block vector is obtained according to a block vector difference (BVD).

[0213] 2. A block vector list is constructed. Candidate block vectors are checked and added in sequence, where AR-BVP candidates are added multiple times. For example, construction can be performed in the following order:

a) a block vector inherited from a neighbouring block;

b) a block vector inherited from a non-neighbouring block;

c) AR-BVP constructed based on existing block vectors;

d) a block vector inherited from a history-based block vector buffer list;

e) AR-BVP constructed based on existing block vectors;

f) a block vector obtained by averaging existing block vectors;

g) a default block vector (related to a block size).

[0214] After adding the block vector for the neighbouring block/the block vector for the non-neighbouring block, AR-BVP construction is implemented. Then, after adding the block vector in the history-based block vector buffer list, AR-BVP construction is implemented.

[0215] 3. A process of AR-BVP construction is determined. Guiding block vectors for AR-BVP construction can include: all block vectors already in the list, some (one or more) block vectors in the list, and a newly constructed block vector (AR-BVP). After a certain guiding block vector is determined, one or more AR-BVPs can be constructed. A termination condition for the construction process can include: the number of iterations of AR-BVP construction, the number of AR-BVPs already added, and the current number of the list. Processes of different AR-BVP construction can be different. For example, during the first AR-BVP construction, all block vectors already in the list are used as guiding block vectors, and the construction is not based on AR-BVPs already added. During the second AR-BVP construction, all block vectors already in the list and a newly added AR-BVP are used as guiding block vectors, and each guiding block vector is used to construct multiple AR-BVPs.

[0216] 4. For a given guiding block vector BV_0 and a block vector BV_ref for a reference block, besides summing BV_0 and BV_ref to obtain BV_arbvp = BV_0 + BV_ref, BV_ref can also be directly used as a candidate block vector, that is, BV_ext = BV_ref, which is then checked and added to the block vector list. For example, when constructing AR-BVP,

after obtaining a block vector for a reference block according to a guiding block vector each time, BV_arbvp and BV_ext can be checked and added.

[0217] 5. A block vector list is constructed according to the above steps, and then encoding of the current block is completed based on the list. For example, in an IBC merge mode, a block vector is selected from the list according to a parsed merge index, and a reference block for the current block is obtained as a prediction block. A reconstructed block for the current block is obtained in combination with parsed residual information.

[0218] When constructing the block vector list, the number of block vectors (HMVP candidates) in an HMVP buffer to be added can be adjusted according to the number of block vectors already added. For example, a maximum length of the block vector list for IBC is *N*, and before adding HMVP candidates, there are *n* candidates already in the list. Then, at most (*N* – *n*)/2 HMVP candidates are added.

[0219] It should be noted that, experimental test is conducted on the decoding method proposed in embodiments of the disclosure. When multi-time AR-BVP construction is applied in the IBC mode, a Y-component coding gain of approximately -0.01% can be achieved on an enhanced compression model (ECM) reference software test platform.

[0220] It should be noted that, experimental test is conducted on the decoding method proposed in embodiments of the disclosure. When a block vector for a reference block is directly used as a candidate block vector in the IntraTMP mode, a Y-component coding gain of approximately -0.03% can be achieved.

[0221] The decoding method proposed in embodiments of the disclosure is exemplarily described below by taking the DIMD mode and the first syntax flag information being cu_dimd_flag as an example. In the DIMD mode, a block vector list of subsample precision can be constructed, and a subsample interpolation-based prediction block is obtained according to block vectors in the block vector list and then fused with prediction blocks obtained through intra prediction to obtain a final prediction block. For example, the subsample interpolation-based prediction block obtained is fused with prediction blocks generated in intra modes derived from HoG.

[0222] Decoding process:

[0223] 1. A flag cu_dimd_flag for a current block is parsed. When cu_dimd_flag is true, it indicates that the DIMD prediction mode is to be applied to a current coding block. For the current block, a HoG can be obtained according to a reconstructed region (or encoded information) and thus intra angular modes can be derived, so as to generate corresponding prediction blocks.

[0224] 2. A block vector list is constructed, where the precision of candidate block vectors is subsample precision. For example, the precision of a block vector is 1/16 samples. The candidate block vectors can be obtained from a neighbouring block, a non-neighbouring block, an AR-BVP candidate, a history-based block vector buffer, etc.

[0225] 3. Template error values for *N* block vectors in the block vector list are calculated. For example, the width of a template is Tw, and the height of the template Th. The width of the current block is W, and the height of the current block is H. A template region is illustrated in FIG. 15.

[0226] The candidate block vector is BV, and prediction on a top template and prediction on a left template are derived. For example, the top template region is obtained. Adjustment is performed on BV according to the height Th of the template, BV_top = BV + (Th << mv_bits), where mv_bits is a subsample bit number. Taking a top-left corner of the current block as a reference, a reference block with a width of W and a height of Th is obtained according to BV_top. If the reference block is unavailable, BV_top is adjusted to be equal to BV and a reference block is obtained. If BV includes a subsample part, a reference block is obtained based on subsample interpolation. An interpolation filter can be a 2-tap interpolation filter, which can be the same as a 2-tap interpolation filter used in inter prediction or IBC prediction. The obtained reference block is used as the prediction on the top template region. The prediction on the left template region is derived in a similar way. Template error values, such as SAD values, are calculated according to prediction values for a template region and reconstructed values for the template region.

[0227] 4. A template error value in a predefined intra non-angular mode can be calculated. For example, one mode Planar is predefined, and a template error value in the Planar mode is calculated according to reference samples.

[0228] 5. An optimal non-angular prediction method is selected according to the candidate block vectors and the template error value in the intra non-angular mode. If the optimal prediction method is to use a certain candidate block vector, in a case that the block vector is of subsample precision, a reference block for the current block can be obtained by means of interpolation filtering, where an interpolation filter can adopt an *n*-tap filter, for example, *n* = 8, which can be the same as an interpolation filter used in IBC prediction.

[0229] 6. According to the above steps, the optimal non-angular prediction method is obtained, and a prediction block generated in the optimal non-angular prediction method is fused with prediction blocks generated in intra angular modes derived from HoG, to obtain a final prediction block, thereby completing encoding of the current block.

[0230] It should be noted that, an experimental test is conducted on the decoding method proposed in embodiments of the disclosure. When a block vector list of subsample-precision is applied in the DIMD mode or the SGPM mode, a Y-component coding gain of approximately -0.01% can be achieved.

[0231] The decoding method proposed in embodiments of the disclosure is exemplarily described below by taking the IntraTMP mode, the first syntax flag information being cu_tmp_flag, the second syntax flag information being cu_tmp_lic_flag, and the third syntax flag information being cu_tmp_idx as an example. A block vector list can be reordered according to template errors. Template region selection and template-error calculation can be implemented in different methods. For example, the template region and the error calculation method for the block vector list can be predefined, or can be consistent with a current prediction process. Taking the IntraTMP mode as an example, for the current block, error calculation methods such as SAD and MR-SAD can be used, and a template region can be classified into an L-shaped template, a top template, and a left template. Corresponding syntax elements in a block vector list-based prediction process are as follows:

```
...
cu_tmp_flag
if(cu_tmp_flag)
{
```

...

cu_tmp_lic_flag

...

cu_tmp_idx

...

}

...

[0232] Decoding process:

[0233] 1. A flag cu_tmp_flag for a current block is parsed. When cu_tmp_flag is true, it indicates that the IntraTMP prediction mode is to be applied to a current coding block. A flag cu_tmp_lic_flag is parsed. When cu_tmp_lic_flag is true, SAD is to be used to calculate template error values for the current block; when cu_tmp_lic_flag is false, MR-SAD is to be used to calculate template error values for the current block. An index value cu_tmp_idx is parsed, and which template types are to be used for template matching is determined according to the value.

[0234] 2. A block vector list is constructed. Candidate block vectors can be obtained from a neighbouring block, a non-neighbouring block, an AR-BVP candidate, a history-based block vector buffer, etc.

[0235] 3. Template error values for candidate block vectors in the block vector list are calculated, and the block vector list is reordered based on the template error values and the first *M* candidates are retained. A template error calculation method to be used and a template to be used are determined according to cu_tmp_lic_flag and cu_tmp_idx. For example, when cu_tmp_lic_flag is true, MR-SAD is to be used to calculate template errors for the current block. When cu_tmp_idx is greater than a threshold *N*, a search result corresponding to an L-shaped template, a search result corresponding to a top template, and a search result corresponding to a left template need to be maintained. Accordingly, MR-SAD is to be used to calculate template errors for the candidates in the block vector list, and template error values diff0 corresponding to the L-shaped template, template error

values diff1 corresponding to the top template, and template error values diff2 corresponding to the left template are obtained. According to diff0, diff1, and diff2 for the candidates, the best *M*0 candidate block vectors corresponding to the L-shaped template, the best *M*1 candidate block vectors corresponding to the top template, and the best *M*2 candidate block vectors corresponding to the left template are obtained, which can be denoted as mrgList0, mrgList1, and mrgList2.

[0236] 4. A maximum template error value pDiff0 corresponding to the L-shaped template, a maximum template error value pDiff1 corresponding to the top template, and a maximum template error value pDiff2 corresponding to the left template can be obtained according to the template errors for the block vector list, and these values are related to the template error values for the candidates in the block vector list. For example, pDiff0 can be equal to the maximum of diff0 among the candidates in mrgList0 plus 1. During subsequent sparse search and refinement search, template error values for candidate block vectors found need to be less than pDiff.

[0237] 5. Sparse search is performed, and search results are recorded as sparseList0, sparseList1, and sparseList2, where sparseList0 is a search result for template error values corresponding to the L-shaped template, sparseList1 is a search result for template error values corresponding to the top template, and sparseList2 is a search result for template error values corresponding to the left template.

[0238] 6. The block vector lists are merged with the sparse search results respectively. For example, mrgList0 and sparseList0 are merged, i.e., the first *K* candidate block vectors with the smallest MR-SAD corresponding to the L-shaped template are retained, where *K* can be an integer greater than or equal to a maximum length of sparseList0. The merged list is denoted as sparseList0'.

[0239] 7. Refinement search is performed on sparseList0', sparseList1', and sparseList2' respectively that are obtained through the above steps, to determine a final IntraTMP block vector candidate list. The final candidate list can be obtained by merging sparselist0', sparselist1', and sparselist2'.

[0240] 8. Based on the final IntraTMP block vector candidate list and parsed cu_tmp_idx, cu_tmp_lic_flag, as well as other related syntax elements, a prediction block for the current block is obtained, thereby completing encoding of the current block.

[0241] It should be noted that, in embodiments of the disclosure, in the IntraTMP mode, SATD and MR-SATD can be used for template-error calculation. In an implementation, SAD is used in a sparse search stage and SATD is used in a refinement search stage. If cu_tmp_lic_flag for the current block is true, MR-SAD and MR-SATD are used respectively. SATD or MR-SATD can be used when calculating template error values for the block vector list, which is consistent with the refinement search stage. In another implementation, whether SATD is to be used for search is indicated via a flag cu_tmp_satd_flag signalled into a bitstream. For example, SAD is to be used when cu_tmp_satd_flag is false, and SATD is to be used when cu_tmp_satd_flag is true. A method for calculating template error values for the block vector list can be determined according to cu_tmp_satd_flag. SAD is to be used when cu_tmp_satd_flag is false, and SATD is to be used when cu_tmp_satd_flag is true. Alternatively, both cu_tmp_satd_flag and cu_tmp_lic_flag can be encoded to determine which one of SAD, SATD, MR-SAD, or MR-SATD is to be used for search and calculation of template errors for the block vector list.

[0242] It should be noted that, in embodiments of the disclosure, a candidate list sparseList obtained through sparse search can be used as guiding block vectors to construct AR-BVP candidates, and template error values for these AR-BVP candidates are calculated and sparseList is updated (*n* candidates with the smallest template error values are retained in the list). AR-BVP candidate construction can be performed for sparseList corresponding to different template types, and sparseList is updated according to template error values for AR-BVP candidates.

[0243] It should be noted that, in embodiments of the disclosure, when the block vector list is used for a search process in the IntraTMP mode, the block vector list can be used only for a search corresponding to one template type. For example, in Embodiment 3, a block vector list mrgList is constructed and reordered according to an L-shaped template, and a candidate list sparseList0 corresponding to the L-shaped template, a candidate list sparseList1 corresponding to the top template, and a candidate list sparseList2

corresponding to the left template are obtained through sparse search. mrgList can be merged with only sparseList0 (the candidate list corresponding to the L-shaped template).

[0244] It should be noted that, an experimental test is conducted on the decoding method proposed in embodiments of the disclosure. When the block vector list is reordered by using SAD and MR-SAD respectively in the IntraTMP mode, a Y-component coding gain of approximately -0.02% can be achieved.

[0245] Embodiments of the disclosure provide a coding method. If the preset intra prediction mode is to be applied to the current block, the block vector list for the current block is constructed; and the prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes the block vector for the reference block corresponding to the guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in the preset template-error calculation method, and the preset template-error calculation method is determined according to the error calculation method for the template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream. That is, in embodiments of the disclosure, the construction and/or usage process for the block vector list during intra prediction can be adjusted and improved. By constructing AR-BVP candidate block vectors multiple times, and/or directly using a block vector for a reference block as a candidate block vector to be added to the block vector list, and/or using a subsample-precision block vector list for intra prediction, and/or reordering the block vector list in different template-error calculation methods, diversity of the block vector list can be improved, thereby improving intra prediction efficiency and improving coding performance.

[0246] In yet another embodiment of the disclosure, FIG. 16 is a schematic flowchart of an encoding method provided in embodiments of the disclosure. As illustrated in FIG. 16, the method can include the following.

[0247] Step 1601, when it is determined that a preset intra prediction mode is to be applied to a current block, first syntax flag information is signalled into a bitstream; where the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block.

[0248] It should be noted that, in embodiments of the disclosure, the method is applied to an encoder. Specifically, based on the structure of the encoder 200 illustrated in FIG. 15, the encoding method of embodiments of the disclosure can be applied to an intra prediction part, and is mainly applied to an intra prediction method based on block vector list construction, which can improve prediction efficiency of intra prediction.

[0249] It should also be noted that, in embodiments of the disclosure, the first syntax flag information indicates whether the preset intra prediction mode is to be applied to the current block. In some embodiments, if a value of the first syntax flag information is a first value, it is determined that the first syntax flag information indicates to apply the preset intra prediction mode to the current block; if the value of the first syntax flag information is a second value, it is determined that the first syntax flag information indicates not to apply the preset intra prediction mode to the current block.

[0250] It should also be noted that, in embodiments of the disclosure, the preset intra prediction mode can include at least one of: IBC, IntraTMP, SGPM, DIMD, or TIMD.

[0251] It can be understood that, in embodiments of the disclosure, the first syntax flag information can be represented differently with regard to different preset intra prediction modes. For example, for IBC, the corresponding first syntax flag information can be represented as cu_pred_mode; for DIMD, the corresponding first syntax flag information can be represented as cu_dimd_flag; for IntraTMP, the corresponding first syntax flag information can be represented as cu_tmp_flag.

[0252] Here, the first value is different from the second value, and the first value and the second value each can be in a parameter or a numerical form. Specifically, the first

syntax flag information can be a parameter written in a profile, or can be a value of a flag, which is not specifically limited herein. Exemplarily, the first value can be 1 and the second value can be 0; alternatively, the first value can be 0 and the second value can be 1; alternatively, the first value can be true and the second value can be false; alternatively, the first value can be false and the second value can be true.

[0253] It can be understood that, in embodiments of the disclosure, after it is determined that the preset intra prediction mode is to be applied to the current block, the corresponding first syntax flag information can be determined, and the first syntax flag information is signalled into the bitstream and transmitted to an encoding end, so that the encoding end determines the preset intra prediction mode to be applied to the current block according to the parsed first syntax flag information.

[0254] It should be noted that, in embodiments of the disclosure, the preset intra prediction mode can be a mode in which intra prediction is performed by constructing a block vector list.

[0255] In some embodiments, the preset intra prediction mode can include at least one of: IBC, IntraTMP, SGPM, DIMD, or TIMD.

[0256] In IBC, prediction is implemented based on a block vector directed to a reference block from the current block. In IntraTMP, a reconstructed block corresponding to the most similar template is determined as a matching block through template matching, which is then used as a prediction block for a current coding block. In SGPM, weighted fusion can be performed on two intra prediction blocks, where weight values can correspond to different GPMs. In DIMD, multiple intra angular modes are derived from HoG, and an optimal non-angular mode is derived by constructing a block vector list and calculating template errors, and weighted fusion is performed on prediction blocks generated in these modes to obtain a final prediction block. In TIMD, an MPM list is constructed according to existing coding information, and prediction is performed on a template region by applying modes in the list, to obtain corresponding template error values, thereby deriving an optimal intra mode.

[0257] Step 1602, a block vector list for the current block is constructed. The block vector list for the current block can satisfy at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream.

[0258] It should be noted that, in embodiments of the disclosure, when it is determined that the preset intra prediction mode is to be applied to the current block, the block vector list for the current block can be constructed.

[0259] Further, in embodiments of the disclosure, the block vector list for the current block can satisfy at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes the block vector for the reference block corresponding to the guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting the template error values corresponding to the block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to the error calculation method for the template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream.

[0260] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be constructed by performing an AR-BVP process at least twice, and between at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. When constructing the block vector list for the current block, different candidate block vectors can be obtained through at least two AR-BVP processes, and the obtained candidate block vectors are checked and then added to the block vector list. Between at least two AR-BVP processes, checking can be performed on at least one non-AR-BVP block vector.

[0261] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can include the block vector for the reference block corresponding to the guiding block vector for the current block. When constructing the block vector list for the current block, the block vector for the reference block determined based on the guiding block vector during an AR-BVP process can be directly used as a candidate block vector, and the candidate block vector is checked and then added to the block vector list.

[0262] It should be noted that, in embodiments of the disclosure, when the preset intra prediction mode is DIMD, or TIMD, or SGPM, the constructed block vector list for the current block can include a subsample-precision block vector. A block vector added to the block vector list for the current block can be of subsample precision, so that a prediction block of subsample precision can be obtained.

[0263] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be obtained by sorting the template error values corresponding to the block vectors in the block vector list. The preset template-error calculation method for calculating the template error values for the block vectors in the block vector list can be determined. The preset template-error calculation method can be determined according to the error calculation method for the template search process corresponding to the current block, or can be determined according to the second syntax flag information transmitted in the bitstream.

[0264] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector(s) in the block vector list for the current block can be determined as a first guiding block vector for the current block, and a first candidate

block vector for the current block can be determined according to the first guiding block vector; where the first block vector includes at least one of: a block vector(s) for a neighbouring block(s) of the current block, a block vector(s) for a non-neighbouring block(s) of the current block, or a block vector(s) in an HMVP buffer. Then, the first candidate block vector and a second block vector(s) are added to the block vector list, where the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. Next, a third block vector in the block vector list is determined as a second guiding block vector for the current block, and a second candidate block vector for the current block is determined according to the second guiding block vector, where the third block vector at least includes the first candidate block vector. The second candidate block vector can be added to the block vector list.

[0265] It should be noted that, in embodiments of the disclosure, the block vector list for the current block can be constructed by using any number (that is, quantity) of block vectors among the block vector for the neighbouring block of the current block, the block vector for the non-neighbouring block of the current block, and the block vector in the HMVP buffer. The first block vector can be any one or more block vectors in the block vector list for the current block, i.e., the first block vector can be a non-AR-BVP block vector.

[0266] Accordingly, in embodiments of the disclosure, the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. The second block vector can be any number of block vectors among the block vector for the neighbouring block of the current block, the block vector for the non-neighbouring block of the current block, and the block vector in the HMVP buffer that are different from the first block vector.

[0267] Accordingly, in embodiments of the disclosure, the third block vector can include the first candidate block vector constructed through an AR-BVP process, and can further include the first block vector and/or the second block vector. That is, after the first candidate block vector and the second block vector are added to the block vector list for the current block, besides the first candidate block vector, other block vectors in the block vector list for the current block can also serve as guiding block vectors for the next AR-BVP process.

[0268] It can be understood that, in embodiments of the disclosure, taking constructing the block vector candidate list for the current block by performing an AR-BVP process twice as an example, during a first AR-BVP process, the first block vector in the block vector list for the current block can be used as the first guiding block vector, and then the corresponding first candidate block vector is determined according to the first guiding block vector, where the first candidate block vector is an AR-BVP block vector. After completing the first AR-BVP process, the obtained first candidate block vector (AR-BVP block vector) and the second block vector (a non-AR-BVP block vector different from the first block vector) can be added to the block vector list for the current block. Then, a second AR-BVP process is performed. The third block vector in the block vector list can be taken as the second guiding block vector, where the third block vector at least includes the first candidate block vector obtained from the first AR-BVP process. Then, the corresponding second candidate block vector can be determined according to the second guiding block vector, where the second candidate block vector is an AR-BVP block vector. Finally, the obtained second candidate block vector can be added to the block vector list.

[0269] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, more than two AR-BVP processes can also be implemented, which is not specifically limited in the disclosure. For example, after the obtained second candidate block vector is added to the block vector list, a guiding block vector for the next AR-BVP process can be determined according to the block vector list, and then a corresponding candidate block vector is determined according to the guiding block vector and added to the block vector candidate list.

[0270] Exemplarily, in some embodiments, it is assumed that when constructing the block vector list, AR-BVPs can be constructed twice. For example, candidate block vectors can be checked and added in the following order:

a) a block vector inherited from a neighbouring block;

b) a block vector inherited from a non-neighbouring block;

c) AR-BVP constructed based on existing block vectors;

d) a block vector inherited from a history-based block vector buffer list;

e) AR-BVP constructed based on existing block vectors;

f) a block vector obtained by averaging existing block vectors;

g) a default block vector.

[0271] The existing block vectors are some or all of the block vectors in the constructed block vector list. After adding the block vector for the neighbouring block/the block vector for the non-neighbouring block, perform AR-BVP for the first time. Thereafter, after adding the block vectors in the history-based block vector buffer list, perform AR-BVP for the second time.

[0272] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector in the block vector list for the current block can be determined as a first guiding block vector for the current block, and a first reference block corresponding to the first guiding block vector is determined, where the first block vector includes at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer. Then, a block vector for the first reference block is determined as a first candidate block vector for the current block, and/or the first guiding block vector and the block vector for the first reference block are summed to determine the first candidate block vector for the current block. Finally, the first candidate block vector can be added to the block vector list.

[0273] It should be noted that, in embodiments of the disclosure, when constructing the block vector list for the current block, a corresponding reference block can be determined according to a guiding block vector, and then a corresponding candidate block vector is determined according to a block vector for the reference block. After completing an AR-BVP process, when determining the candidate block vector according to the block vector for the reference block, a result obtained by summing the guiding block vector and the block vector for the reference block can be determined as the candidate block vector and added to the block vector candidate list, or the block vector for the reference block can be directly determined as the candidate block vector and added to the block vector candidate list.

[0274] It can be understood that, in embodiments of the disclosure, when determining the first candidate block vector according to the first guiding block vector, the first reference

block can be determined according to the first guiding block vector, and then the first candidate block vector is determined according to the block vector for the first reference block. The first candidate block vector can be determined by summing the first guiding block vector and the block vector for the first reference block, or the block vector for the first reference block can be directly determined as the first candidate block vector. Alternatively, the first candidate block vector can include both a result obtained by summing the first guiding block vector and the block vector for the first reference block, and the block vector for the first reference block.

[0275] That is, in embodiments of the disclosure, a block vector for a reference block determined according to a guiding block vector can be directly added to the block vector list as a candidate block vector, i.e., the block vector list for the current block includes the block vector for the reference block corresponding to the guiding block vector for the current block.

[0276] Exemplarily, in some embodiments, it is assumed that when constructing the block vector list, the block vector for the reference block can be directly used as a block vector candidate (candidate block vector). For a given guiding block vector BV_0 and a block vector BV_ref for a corresponding reference block, a candidate block vector BV_arbvp can be obtained by summing the guiding block vector and the block vector for the reference block, i.e., BV_arbvp = BV_0 + BV_ref, or the block vector BV_ref for the reference block can be directly used as a candidate block vector, i.e., BV_ext = BV_ref. Alternatively, when constructing AR-BVP, after obtaining the vector for the reference block according to the guiding block vector each time, both BV_arbvp and BV_ext can be added to the block vector list as candidate block vectors.

[0277] Further, in embodiments of the disclosure, when constructing the block vector list for the current block, a first block vector in the block vector list for the current block can be determined as a first guiding block vector(s) for the current block, and a first reference block corresponding to the first guiding block vector is determined, where the first block vector includes at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer. Then, a block vector for the first reference block is determined as a first candidate block vector for the current block, and/or the first guiding block vector and the

block vector for the first reference block are summed to determine the first candidate block vector for the current block. Next, the first candidate block vector and a second block vector are added to the block vector list, where the second block vector can be a non-AR-BVP block vector, for example, the second block vector can be a non-AR-BVP block vector different from the first block vector. Then, a third block vector in the block vector list is determined as a second guiding block vector(s) for the current block, and a second reference block corresponding to the second guiding block vector is determined, where the third block vector at least includes the first candidate block vector. Next, a block vector for the second reference block is determined as a second candidate block vector for the current block, and/or the second guiding block vector and the block vector for the second reference block are summed to determine the second candidate block vector for the current block. Finally, the second candidate block vector can be added to the block vector list.

[0278] It can be understood that, in embodiments of the disclosure, taking constructing the block vector candidate list for the current block through two AR-BVP processes as an example, during a first AR-BVP process, the first block vector in the block vector list for the current block can be used as the first guiding block vector, and then the corresponding first candidate block vector is determined according to the first guiding block vector. The first candidate block vector is an AR-BVP block vector, and the first candidate block vector can include the block vector for the first reference block corresponding to the first candidate block vector, and/or a result obtained by summing the first guiding block vector and the block vector for the corresponding first reference block. After completing the first AR-BVP process, the obtained first candidate block vector (AR-BVP block vector) and the second block vector (a non-AR-BVP block vector different from the first block vector) can be added to the block vector list for the current block. Then, a second AR-BVP process is performed. The third block vector in the block vector list can be determined as the second guiding block vector, where the third block vector at least includes the first candidate block vector obtained from the first AR-BVP process. Then, the corresponding second candidate block vector can be determined according to the second guiding block vector. The second candidate block vector is an AR-BVP block vector, and the second candidate block vector can include the block vector for the second reference block corresponding to the second candidate block vector, and/or a result obtained by summing the second guiding block vector and the block vector

for the corresponding second reference block. Finally, the obtained second candidate block vector can be added to the block vector list.

[0279] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, more than two AR-BVP processes can also be implemented, which is not specifically limited in the disclosure. For example, after the obtained second candidate block vector is added to the block vector list, a guiding block vector for a next AR-BVP process can be determined according to the block vector list, and then a corresponding candidate block vector is determined according to the guiding block vector and added to the block vector candidate list.

[0280] It can be understood that, in embodiments of the disclosure, when constructing the block vector list for the current block, a block vector for a reference block corresponding to a guiding vector can be added to the block vector list for the current block during any one or more AR-BVP processes.

[0281] That is, in embodiments of the disclosure, when constructing the block vector list, AR-BVP construction can be performed in at least two times. In addition, during an AR-BVP process, a block vector for a reference block can be directly used as a candidate block vector.

[0282] Further, in embodiments of the disclosure, when performing an AR-BVP process to obtain a candidate block vector, a guiding block vector can be determined according to existing block vectors in the block vector list. The first guiding block vector in a first AR-BVP process and the second guiding block vector in a second AR-BVP process can be partially the same or completely different.

[0283] That is, in embodiments of the disclosure, processes of AR-BVP construction can be the same or different. For example, during the first AR-BVP construction, all existing block vectors in the block vector list can be used as guiding block vectors; during the second AR-BVP construction, all existing block vectors in the list and newly added AR-BVP can be used as guiding block vectors.

[0284] Exemplarily, in some embodiments, the first guiding block vector in the first AR-BVP process can include some or all of the block vectors in the block vector list. The second

guiding block vector in the second AR-BVP process can include a non-AR-BVP block vector in the block vector list that is different from the first guiding block vector, and can further include a newly constructed AR-BVP block vector (i.e., the first candidate block vector) added to the block vector list after the first AR-BVP process.

[0285] Further, in embodiments of the disclosure, an AR-BVP process(es) can be performed according to the guiding block vector for the current block to obtain at least one candidate block vector, so as to construct the block vector list for the current block, until a preset termination condition is satisfied, where the preset termination condition includes at least one of: the number of AR-BVP processes is greater than or equal to a preset threshold, the number of candidate block vectors determined through AR-BVP processes is greater than or equal to a first number threshold, or the number of block vectors in the block vector list is greater than or equal to a second number threshold.

[0286] It can be understood that, in embodiments of the disclosure, the preset threshold, the first number threshold, and the second number threshold each can be an integer greater than 0 set in advance, which is not specifically limited in the disclosure. For example, the preset threshold can be 3, the first number threshold can be 5, and the second number threshold can be 15.

[0287] That is, in embodiments of the disclosure, after a guiding block vector (e.g., the first guiding block vector or the second guiding block vector) is determined, one or more AR-BVP block vectors can be constructed. For example, a reference block B_1 is determined according to a guiding block vector $BV_{0,1}$, and then a new block vector $BV_{0,2} = BV_{0,1} + BV_{1,2}$ is constructed according to a block vector $BV_{1,2}$ existing in a region of the reference block B_1. Then, AR-BVP process can be continued, e.g., a reference block B_n can be determined according to $BV_{0,n}$, and if there is a block vector $BV_{n,n+1}$ in a region of B_n, a new block vector $BV_{0,n+1} = BV_{0,1} + BV_{1,2} + \cdots + BV_{n,n+1}$ can be constructed.

[0288] It should be noted that, in embodiments of the disclosure, the termination condition (preset termination condition) for the AR-BVP construction process can include but is not limited to: the number of iterations of AR-BVP construction, the number of AR-

BVPs already added to the block vector list, and the current number of block vectors in the block vector list.

[0289] It should be noted that, in embodiments of the disclosure, processes of AR-BVP construction can be the same or different. For example, during the first AR-BVP construction, construction may not be continued based on already added AR-BVPs (the first candidate block vectors); during the second AR-BVP construction, each guiding block vector can be used to construct multiple AR-BVPs, i.e., construction can be continued based on a newly added AR-BVP (the second candidate block vector).

[0290] Further, in embodiments of the disclosure, when determining a candidate block vector for the current block according to a block vector for a reference block, summation is performed on the guiding block vector and a block vector for at least one reference block respectively to determine the candidate block vector; and/or summation is performed on the guiding block vector and at least one block vector for the reference block respectively to determine the candidate block vector.

[0291] It should be noted that, in embodiments of the disclosure, one or more corresponding reference blocks can be determined according to a guiding block vector, and a block vector(s) for the one or more corresponding reference blocks can also be determined according to the guiding block vector.

[0292] Exemplarily, in some embodiments, when determining the first candidate block vector for the current block according to the block vector for the first reference block, summation is performed on the first guiding block vector and a block vector for at least one first reference block respectively to determine the first candidate block vector; and/or summation is performed on the first guiding block vector and at least one block vector for the first reference block respectively to determine the first candidate block vector.

[0293] That is, in embodiments of the disclosure, for one guiding block vector, at least one corresponding reference block can be determined at at least one position in a corresponding reference region. For any one of the reference blocks, the reference block can correspond to at least one block vector.

[0294] Exemplarily, in some embodiments, as illustrated in FIG. 14, when determining a block vector corresponding to a reference block, multiple positions can be checked. For example, the center, top-left, top-right, bottom-left, and bottom-right of the reference block are checked in sequence to determine whether there is a coding block, and if there is such block, a block vector stored for the block is taken as the block vector. Any one reference block may correspond to one or more block vectors.

[0295] Further, in embodiments of the disclosure, the number of block vectors in the HMVP buffer to be added to the block vector list can be determined according to the number of block vectors in the block vector list.

[0296] That is, in embodiments of the disclosure, when constructing the block vector list, the number of block vectors in the HMVP buffer to be added to the block vector list can be adjusted according to the number of block vectors already added to the block vector list.

[0297] Exemplarily, in some embodiments, assuming that a maximum length of the block vector list is *N*, and there are already *n* block vectors in the block vector list before adding block vectors in the HMVP buffer, then the number of block vectors in the HMVP buffer to be added to the block vector list can be set according to *N* and *n*. For example, the number of block vectors in the HMVP buffer to be added to the block vector list is determined to be (*N* - *n*)/2.

[0298] Further, in embodiments of the disclosure, before adding any block vector (an AR-BVP block vector or a non-AR-BVP block vector) to the block vector list, the block vector is checked. If the block vector does not satisfy a preset addition condition, the block vector will not be added to the block vector list; if the block vector satisfies the preset addition condition, the block vector will be added to the block vector list. The preset addition condition includes at least one of: the block vector is different from any block vector in the block vector list; or a reference block corresponding to the block vector exists.

[0299] Exemplarily, in some embodiments, between at least two AR-BVP processes, at least one non-AR-BVP block vector is checked. For example, a block vector (non-AR-BVP block vector) in the HMVP buffer is checked to determine whether the block vector in the HMVP buffer can be added to the block vector list for the current block. If the block vector

in the HMVP buffer is the same as a block vector already added to the list, the non-AR-BVP block vector will not be added, i.e., for a non-AR-BVP block vector checked, if the addition condition is not satisfied, the non-AR-BVP block vector will not be added.

[0300] Accordingly, in some embodiments, between at least two AR-BVP processes, after at least one non-AR-BVP block vector is checked, if a certain non-AR-BVP block vector is different from all block vectors already added to the list and a reference block corresponding to the non-AR-BVP block vector exists, it can be determined that the non-AR-BVP block vector can be added to the block vector list for the current block.

[0301] Step 1603, a prediction block for the current block is determined according to the block vector list for the current block.

[0302] In embodiments of the disclosure, after the block vector list for the current block is constructed, the prediction block for the current block can be determined according to the block vector list for the current block.

[0303] Further, in embodiments of the disclosure, when the preset intra prediction mode is DIMD, or TIMD, or SGPM, the constructed block vector list for the current block can include a subsample-precision block vector. Accordingly, when determining the prediction block for the current block according to the block vector list for the current block, template error values for block vectors in the block vector list can be determined according to the subsample-precision block vector in the block vector list; and then the prediction block for the current block can be determined according to the template error values for the block vectors in the block vector list.

[0304] It can be understood that, in embodiments of the disclosure, for a block vector list of subsample precision, each block vector in the block vector list corresponds to a subsample precision parameter, which can characterize a precision value of the block vector. The subsample precision parameter can be a value greater than 0 and less than or equal to 1, for example, the subsample precision parameter for a block vector is 1/16.

[0305] Further, in embodiments of the disclosure, when determining the template error values for the block vectors in the block vector list according to the subsample-precision block vector in the block vector list, a template type parameter and a template size parameter

corresponding to the template type parameter can be determined; then, an adjusted block vector corresponding to the subsample-precision block vector is determined according to the template type parameter, the template size parameter, and a subsample bit parameter for the subsample-precision block vector; prediction values for a reference template are determined according to the adjusted block vector; and the template error values for the block vectors in the block vector list can be determined according to the prediction values for the reference template and reconstructed values for a current template.

[0306] It should be noted that, in embodiments of the disclosure, the template type parameter can be used to determine the type of a template. According to the template type parameter, the type of a corresponding template can be determined to be a top template or a left template.

[0307] It should be noted that, in embodiments of the disclosure, the template size parameter can be used to determine the size of a template. According to the template size parameter, the width or height of a corresponding template can be determined. For a top template, the corresponding width is a width value of the current block, and the corresponding height can be determined according to the template size parameter; for a left template, the corresponding height is a height value of the current block, and the corresponding width can be determined according to the template size parameter.

[0308] Exemplarily, in some embodiments, as illustrated in FIG. 15, the current block has a height value H and a width value W. Two template types, such as a top template and a left template, corresponding to the current block are determined according to the template type parameter. Based on the template size parameter for the top template, the width of the top template can be determined to be Tw; based on the template size parameter for the left template, the height of the left template can be determined to be Th.

[0309] It should be noted that, in embodiments of the disclosure, when determining the adjusted block vector corresponding to the subsample-precision block vector according to the template type parameter, the template size parameter, and the subsample bit parameter for the subsample-precision block vector, a template corresponding to the current block and a size value of the template can be determined according to the template type parameter and the template size parameter, and then adjustment is performed according to the

subsample bit parameter corresponding to the block vector and the size value of the template, to obtain the adjusted block vector.

[0310] Exemplarily, in some embodiments, assuming that the template corresponding to the current block is determined to be a top template according to the template type parameter, the size value of the corresponding template is the height Th, and the subsample bit parameter for the corresponding block vector BV is mv_bits, then the block vector BV can be adjusted according to the following formula to obtain the corresponding adjusted block vector BV_top:

$$BV_top = BV + (Th << mv_bits) \quad (1)$$

[0311] Exemplarily, in some embodiments, assuming that the template corresponding to the current block is determined to be a left template according to the template type parameter, the size value of the corresponding template is the width Tw, and the subsample bit parameter for the corresponding block vector BV is mv_bits, then the block vector BV can be adjusted according to the following formula to obtain the corresponding adjusted block vector BV_left:

$$BV_left = BV + (Tw << mv_bits) \quad (2)$$

[0312] Further, in embodiments of the disclosure, when determining the prediction values for the reference template according to the adjusted block vector, if a candidate reference template corresponding to the adjusted block vector is available, the prediction values for the reference template can be determined according to the adjusted block vector and a subsample interpolation algorithm; if the candidate reference template corresponding to the adjusted block vector is unavailable, the prediction values for the reference template can be determined according to block vectors in the block vector list and the subsample interpolation algorithm.

[0313] It can be understood that, in embodiments of the disclosure, a corresponding candidate reference template can be determined based on the adjusted block vector. If the candidate reference template is available, the prediction values for the reference template can be directly determined according to the adjusted block vector and the subsample interpolation algorithm; if the candidate reference template is unavailable, the adjusted block

vector is no longer used, and instead, the prediction values for the reference template are determined according to a block vector before adjustment and the subsample interpolation algorithm.

[0314] Exemplarily, in some embodiments, assuming that the adjusted block vector is BV_top, a candidate reference template with a width of W and a height of Th is determined according to BV_top. If the candidate reference template is available, the prediction values for the corresponding reference template can be calculated based on BV_top by means of the subsample interpolation algorithm; if the candidate reference template is unavailable, BV_top can be adjusted to be equal to BV, and the prediction values for the corresponding reference template can be calculated by means of the subsample interpolation algorithm.

[0315] It should be noted that, in embodiments of the disclosure, the subsample interpolation algorithm can be implemented with an interpolation filter, where the interpolation filter can be a 2-tap interpolation filter, which can be the same as a 2-tap interpolation filter used for inter prediction or IBC prediction.

[0316] Accordingly, in embodiments of the disclosure, prediction on a left template region can also be performed through the above method, i.e., the prediction values for the corresponding reference template is determined according to BV_left. As such, the template error values corresponding to the block vectors can be calculated according to the prediction values for the reference template and the reconstructed values for the current template, and then the prediction block for the current block can be determined according to the template error values for the block vectors in the block vector list.

[0317] Exemplarily, in some embodiments, assuming that the preset intra prediction mode is DIMD, a template error value in a predefined intra non-angular mode can be calculated. For example, one mode Planar is predefined, and a template error value in the Planar mode is calculated according to reference samples. Then, an optimal non-angular prediction method can be selected according to the candidate block vectors in the block vector list and the template error value in the intra non-angular mode. If the optimal prediction method is to use a certain candidate block vector, in a case that the block vector is of subsample precision, a reference block for the current block can be obtained by means of interpolation filtering, where an interpolation filter can adopt an *n*-tap filter, for example, *n*

= 8, which can be the same as an interpolation filter used for IBC prediction. After the optimal non-angular prediction method is determined, a prediction block generated in the optimal non-angular prediction method is fused with prediction blocks generated in intra angular modes derived from HoG, to obtain a final prediction block.

[0318] Further, in embodiments of the disclosure, the block vector list for the current block can be obtained by sorting template error values corresponding to block vectors in the block vector list. Accordingly, when determining the prediction block for the current block according to the block vector list for the current block, block-vector template-error calculation can be performed on block vectors in the block vector list to determine template error values for the block vectors in the block vector list; then, the block vector list can be sorted according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list; and the prediction block for the current block can be determined based on the sorted block vector list.

[0319] It should be noted that, in embodiments of the disclosure, a preset template-error calculation method used for calculating the template error values for the block vectors in the block vector list can be determined. The preset template-error calculation method can be determined according to an error calculation method for a template search process corresponding to the current block, or can be determined according to the second syntax flag information transmitted in the bitstream.

[0320] It can be understood that, in embodiments of the disclosure, in an implementation, the second syntax flag information indicates that the preset template-error calculation method is SAD or MR-SAD, which can be represented by cu_tmp_lic_flag. In some embodiments, if a value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SAD.

[0321] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is SATD or MR-SATD, which can be represented by cu_tmp_lic_flag. In

some embodiments, if the value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SATD.

[0322] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is SATD or SAD, which can be represented by cu_tmp_satd_flag. In some embodiments, if the value of the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD; if the value of the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD.

[0323] It can be understood that, in embodiments of the disclosure, in another implementation, the second syntax flag information indicates that the preset template-error calculation method is one of SAD, MR-SAD, SATD, or MR-SATD, which can be represented jointly by cu_tmp_lic_flag and cu_tmp_satd_flag. In some embodiments, if a value of first information cu_tmp_lic_flag in the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD or SATD; if the value of the first information cu_tmp_lic_flag in the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is MR-SAD or MR-SATD. If a value of second information cu_tmp_satd_flag in the second syntax flag information is a first value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SATD or MR-SATD; if the value of the second information cu_tmp_satd_flag in the second syntax flag information is a second value, it is determined that the second syntax flag information indicates that the preset template-error calculation method is SAD or MR-SAD. For example, if the value of cu_tmp_lic_flag is the first value and the value of cu_tmp_satd_flag is the first value, it can be determined that the preset template-error calculation method is SATD; if the value of

cu_tmp_lic_flag is the second value and the value of cu_tmp_satd_flag is the first value, it can be determined that the preset template-error calculation method is MR-SATD.

[0324] Here, the first value is different from the second value, and the first value and the second value each can be in a parameter or a numerical form. Specifically, the second syntax flag information can be a parameter written in a profile, or can be a value of a flag, which is not specifically limited herein. Exemplarily, the first value can be 1 and the second value can be 0; alternatively, the first value can be 0 and the second value can be 1; alternatively, the first value can be true and the second value can be false; alternatively, the first value can be false and the second value can be true.

[0325] Further, in embodiments of the disclosure, when performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list, a preset template-error calculation method can be determined, and the second syntax flag information is signalled into the bitstream; where the second syntax flag information is used to determine the preset template-error calculation method, and the preset template-error calculation method includes any one of: SAD, MR-SAD, SATD, or MR-SATD. Finally, template error values for the block vectors in the candidate block vector list corresponding to at least one template type can be determined based on the preset template-error calculation method.

[0326] It can be understood that, in embodiments of the disclosure, the preset template-error calculation method for performing block-vector template-error calculation can include any one of SAD, MR-SAD, SATD, or MR-SATD. After the preset template-error calculation method for performing block-vector template-error calculation is determined, the corresponding second syntax flag information can be determined, and the second syntax flag information is signalled into the bitstream and transmitted to the decoding end.

[0327] Exemplarily, in some embodiments, assuming that the second syntax flag information is represented by cu_tmp_lic_flag, then according to a value of cu_tmp_lic_flag, whether the preset template-error calculation method is SAD or is MR-SAD can be determined, or whether the preset template-error calculation method is SATD or is MR-SATD can be determined.

[0328] Exemplarily, in some embodiments, assuming that the second syntax flag information is represented by cu_tmp_satd_flag, then according to a value of cu_tmp_satd_flag, whether the preset template-error calculation method is SAD or is SATD can be determined.

[0329] Exemplarily, in some embodiments, assuming that the second syntax flag information includes first information cu_tmp_lic_flag and second information cu_tmp_satd_flag, then which one of SAD, MR-SAD, SATD, or MR-SATD is the preset template-error calculation method can be determined according to a value of cu_tmp_lic_flag and a value of cu_tmp_satd_flag. For example, if the value of cu_tmp_lic_flag is a first value and the value of cu_tmp_satd_flag is a first value, it can be determined that the preset template-error calculation method is SATD; if the value of cu_tmp_lic_flag is a second value and the value of cu_tmp_satd_flag is a first value, it can be determined that the preset template-error calculation method is MR-SATD.

[0330] Further, in embodiments of the disclosure, when determining the preset template-error calculation method, a first template-error calculation method and a second template-error calculation method can be determined; where the first template-error calculation method is used for performing sparse search on the at least one template type, and the second template-error calculation method is used for performing refinement search on the at least one template type and block-vector template-error calculation.

[0331] That is, in embodiments of the disclosure, the first template-error calculation method used for performing sparse search and the second template-error calculation method used for performing refinement search and block-vector template-error calculation can be determined, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage.

[0332] Further, in embodiments of the disclosure, when determining the preset template-error calculation method, a third template-error calculation method and a fourth template-error calculation method can be determined; where the third template-error calculation method is used for performing sparse search on the at least one template type and block-vector template-error calculation, and the fourth template-error calculation method is used for performing refinement search on the at least one template type.

[0333] That is, in embodiments of the disclosure, the third template-error calculation method used for performing sparse search and block-vector template-error calculation and the fourth template-error calculation method used for performing refinement search can be determined, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a sparse search stage.

[0334] Further, in embodiments of the disclosure, when determining the preset template-error calculation method, a fifth template-error calculation method can be determined; where the fifth template-error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

[0335] That is, in embodiments of the disclosure, the fifth template-error calculation method used for performing sparse search, refinement search, and block-vector template-error calculation can be determined, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage and a sparse search stage.

[0336] Further, in embodiments of the disclosure, the second syntax flag information includes first information and second information. When determining the preset template-error calculation method, a sixth error calculation method is determined; where the sixth error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation. Then, the first information and the second information can be determined according to the sixth error calculation method.

[0337] That is, in embodiments of the disclosure, the sixth template-error calculation method used for performing sparse search, refinement search, and block-vector template-error calculation can be determined, i.e., block-vector template-error calculation is consistent with the template-error calculation method used in a refinement search stage and a sparse search stage.

[0338] That is, in embodiments of the disclosure, both cu_tmp_satd_flag and cu_tmp_lic_flag flags can be encoded to indicate which one of SAD, SATD, MR-SAD, or MR-SATD is to be used for search and template error calculation for a block vector list.

[0339] Further, when performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list, block-vector template-error calculation can be performed on the block vectors in the block vector list according to a preset template-error calculation method for sparse search in a template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list; alternatively, block-vector template-error calculation can be performed on the block vectors in the block vector list according to a preset template-error calculation method for refinement search in the template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list.

[0340] That is, in embodiments of the disclosure, the template-error calculation method for a sparse search stage can be directly used as the preset template-error calculation method for block-vector template-error calculation, or the template-error calculation method for a refinement search stage can be directly used as the preset template-error calculation method for block-vector template-error calculation.

[0341] Further, in embodiments of the disclosure, when sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list, at least one template-number corresponding to the at least one template type can be determined, and third syntax flag information is signalled into the bitstream; where the third syntax flag information is used to determine the at least one template-number corresponding to the at least one template type. Then, the template error values for the block vectors in the candidate block vector list corresponding to the at least one template type are sorted separately according to the at least one template-number, to obtain at least one first candidate list corresponding to the at least one template type.

[0342] It should be noted that, in embodiments of the disclosure, for each of different template types, a template-number corresponding to the template type can be determined. Then, template error values for block vectors corresponding to the template type are sorted

according to the template-number. As such, first candidate lists corresponding to different template types can be obtained, i.e., the sorted block vector list includes first candidate lists corresponding to different template types.

[0343] It can be understood that, in embodiments of the disclosure, the third syntax flag information indicates template-numbers corresponding to different template types, which can be represented by cu_tmp_idx. In some embodiments, according to a value of the third syntax flag information, a search result that needs to be maintained for each of different template types can be determined. For example, it is determined, according to the value of the third syntax flag information, that a search result corresponding to an L-shaped template, a search result corresponding to a top template, and a search result corresponding to a left template need to be maintained, i.e., a template-number corresponding to the L-shaped template is determined to be *M*0, a template-number corresponding to the top template is determined to be *M*1, and a template-number corresponding to the left template is determined to be *M*2, where *M*0, *M*1, and *M*2 each are an integer greater than or equal to 0.

[0344] Exemplarily, in some embodiments, for the current block, MR-SAD is used to calculate template errors. When the value of the third syntax flag information cu_tmp_idx is greater than a threshold *N* (e.g., *N*=13), a search result corresponding to the L-shaped template, a search result corresponding to the top template, and a search result corresponding to the left template need to be maintained. Accordingly, MR-SAD is used to calculate template errors for the block vectors in the block vector list, and template error values diff0 corresponding to the L-shaped template, template error values diff1 corresponding to the top template, and template error values diff2 corresponding to the left template are obtained. According to diff0, diff1, and diff2 for the block vectors, the best *M*0 candidate block vectors corresponding to the L-shaped template, the best *M*1 candidate block vectors corresponding to the top template, and the best *M*2 candidate block vectors corresponding to the left template are obtained, which can be denoted as mrgList0, mrgList1, and mrgList2 respectively, where mrgList0 is a first candidate list corresponding to the L-shaped template, mrgList1 is a first candidate list corresponding to the top template, and mrgList2 is a first candidate list corresponding to the left template.

[0345] Further, in embodiments of the disclosure, when determining the prediction block for the current block based on the sorted block vector list, sparse search can be performed separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively; then, the at least one first candidate list is merged with the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type; next, refinement search can be performed separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; finally, a target candidate list corresponding to the current block can be determined according to the at least one fourth candidate list, and the prediction block for the current block can be determined according to the target candidate list.

[0346] Exemplarily, in some embodiments, sparse search is performed separately for the L-shaped template, the top template, and the left template, and search results are recorded as sparseList0, sparseList1, and sparseList2, where sparseList0 is a search result for template error values corresponding to the L-shaped template, sparseList1 is a search result for template error values corresponding to the top template, and sparseList2 is a search result for template error values corresponding to the left template. Here, sparseList0 is a second candidate list corresponding to the L-shaped template, sparseList1 is a second candidate list corresponding to the top template, and sparseList2 is a second candidate list corresponding to the left template.

[0347] Exemplarily, in some embodiments, the block vector list(s) and the sparse search result(s) are merged, i.e., the at least one first candidate list and the at least one second candidate list are merged respectively. For example, mrgList0 and sparseList0 are merged, i.e., the first *K* candidate block vectors with the smallest MR-SAD corresponding to the L-shaped template are retained, where *K* can be an integer greater than or equal to the maximum length of sparseList0, and the merged list is denoted as sparseList0'. Accordingly, mrgList1 and sparseList1 can be merged to obtain corresponding sparseList1', and mrgList2 and sparseList2 can be merged to obtain corresponding sparseList2'. Here, sparseList0' is a third candidate list corresponding to the L-shaped template, sparseList1' is a third

candidate list corresponding to the top template, and sparseList2' is a third candidate list corresponding to the left template.

[0348] Exemplarily, in some embodiments, based on the third candidate lists obtained by merging, i.e., sparseList0' corresponding to the L-shaped template, sparseList1' corresponding to the top template, and sparseList2' corresponding to the left template, refinement search can be performed respectively to obtain a search result sparselist0' for template error values corresponding to the L-shaped template, a search result sparselist1' for template error values corresponding to the top template, and a search result sparselist2' for template error values corresponding to the left template, where sparselist0' is a fourth candidate list corresponding to the L-shaped template, sparselist1' is a fourth candidate list corresponding to the top template, and sparselist2' is a fourth candidate list corresponding to the left template.

[0349] Exemplarily, in some embodiments, when determining the target candidate list corresponding to the current block according to the at least one fourth candidate list, the at least one fourth candidate list can be merged, for example, sparselist0', sparselist1', and sparselist2' are merged to obtain a final target candidate list.

[0350] Further, in embodiments of the disclosure, after at least one second candidate list is obtained through sparse search, for any one of the second candidate lists, AR-BVP construction can be performed based on the second candidate list. A third guiding block vector can be determined according to the second candidate list, and then a third candidate block vector(s) for the current block is determined according to the third guiding block vector. Next, a template error value corresponding to the third candidate block vector is determined, and the second candidate list is updated according to the template error value corresponding to the third candidate block vector.

[0351] Exemplarily, in some embodiments, a block vector in a candidate list sparseList obtained through sparse search can be used as a guiding block vector (third guiding block vector) to construct AR-BVP candidates (third candidate block vectors), and then template error values for these AR-BVP candidates can be calculated and sparseList can be updated. AR-BVP candidate construction can be performed for sparseList corresponding to different

template types, and sparseList is updated according to template error values for AR-BVP candidates.

[0352] Further, in embodiments of the disclosure, when determining the prediction block for the current block based on the sorted block vector list, sparse search can be performed separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively; then, a first candidate list corresponding to a preset template type is merged with the at least one second candidate list to obtain at least one third candidate list corresponding to the at least one template type; next, refinement search can be performed separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; finally, a target candidate list corresponding to the current block can be determined according to the at least one fourth candidate list, and the prediction block for the current block can be determined according to the target candidate list.

[0353] It should be noted that, in embodiments of the disclosure, when merging a block vector list and sparse search result(s), the first candidate list corresponding to the preset template type can be merged with the at least one second candidate list, to obtain at least one third candidate list corresponding to the at least one template type. The preset template type can be any one of an L-shaped template, a top template, or a left template.

[0354] Exemplarily, in some embodiments, when merging the block vector list and the sparse search result(s), mrgList0 can be merged with each of sparseList0, sparseList1, and sparseList2, and the merged lists are denoted as sparseList0', sparseList1', and sparseList2'. sparseList0' is a third candidate list corresponding to the L-shaped template, sparseList1' is a third candidate list corresponding to the top template, and sparseList2' is a third candidate list corresponding to the left template.

[0355] That is, in embodiments of the disclosure, a block vector list mrgList can be constructed and reordered according to the preset template type (e.g., the L-shaped template), and a candidate list sparseList0 corresponding to the L-shaped template, a candidate list sparseList1 corresponding to the top template, and a candidate list sparseList2

corresponding to the left template can be obtained through sparse search. Subsequently, only mrgList is merged with each of sparseList0, sparseList1, and sparseList2.

[0356] Further, in embodiments of the disclosure, during refinement search based on at least one third candidate list, all block vectors in the third candidate list correspond to the same template search region. Alternatively, during refinement search based on at least one third candidate list, fourth block vectors in the third candidate list correspond to the same template search region, where the fourth block vectors are block vectors derived from the first candidate list.

[0357] It should be noted that, in embodiments of the disclosure, when performing refinement search separately for at least one template type based on at least one third candidate list, template search regions corresponding to block vectors in the third candidate list can be the same or different. For example, two template search regions corresponding to any two block vectors in the third candidate list can be the same; or two template search regions corresponding to any two fourth block vectors in the third candidate list can be the same.

[0358] It can be understood that, in embodiments of the disclosure, the fourth block vectors can be block vectors in the third candidate list that are derived from the first candidate list. That is, when merging the first candidate list and the second candidate list, the fourth block vectors in the first candidate list constitute the third candidate list.

[0359] Further, in embodiments of the disclosure, when determining the prediction block for the current block according to the block vector list for the current block, a target block vector for the current block can be determined from the block vector list for the current block; then, the prediction block for the current block can be determined according to the target block vector. Accordingly, a block vector index ID for the current block can be determined according to the target block vector, and the block vector index ID is signalled into the bitstream.

[0360] Exemplarily, in some embodiments, if the preset intra prediction mode is IBC, the block vector index ID for the current block can be determined according to the target

block vector. For example, in an IBC merge mode, a merge index (block vector index ID) is determined according to the target block vector.

[0361] Further, in embodiments of the disclosure, a residual block for the current block is determined according to an original block of the current block and the prediction block for the current block; the residual block for the current block is signalled into the bitstream.

[0362] It can be understood that, in embodiments of the disclosure, after the prediction block for the current block is determined, a difference between the original block of the current block and the prediction block for the current block can be determined as the residual block for the current block.

[0363] In summary, through the encoding method proposed in the above step 1601 to step 1603, the construction and/or usage process for the block vector list during intra prediction is adjusted and improved. When constructing the block vector list, AR-BVP candidate block vectors can be constructed multiple times. In addition, a block vector for a reference block can also be directly used as a candidate block vector and added to the block vector list. Furthermore, a block vector list of subsample precision can be used for intra prediction. Moreover, various methods can be applied to determine multiple template-error calculation methods, so the block vector list can be reordered by using different template-error calculation methods.

[0364] That is, the encoding method proposed in embodiments of the disclosure is a method for improving construction and/or usage of a block vector list, where the block vector list can be an AMVP/Merge list in an IBC mode, a Merge list in an IntraTMP mode, or a block vector list constructed to obtain block-vector-based prediction in other intra modes. During construction and/or use of the block vector list, diversity of the block vector list and utilization efficiency of the block vector list are improved by applying methods such as constructing AR-BVP candidate block vectors multiple times, and/or directly adding a block vector for a reference block to the block vector list, and/or using a block vector list of subsample precision in intra modes, and/or sorting the block vector list by using different template-error calculation methods.

[0365] The encoding method proposed in embodiments of the disclosure is exemplarily described below by taking the IBC mode and the first syntax flag information being cu_pred_mode as an example. When constructing a block vector list, AR-BVPs can be constructed multiple times, and a block vector for a reference block can also be directly added to the block vector list as a block vector candidate.

[0366] Encoding process:

[0367] 1. A current block is encoded, and it is determined to select the IBC mode for encoding the current block.

[0368] 2. A block vector list is constructed. Candidate block vectors are checked and added in sequence, where AR-BVP candidates are added multiple times. For example, construction can be performed in the following order:

a) a block vector inherited from a neighbouring block;

b) a block vector inherited from a non-neighbouring block;

c) AR-BVP constructed based on existing block vectors;

d) a block vector inherited from a history-based block vector buffer list;

e) AR-BVP constructed based on existing block vectors;

f) a block vector obtained by averaging existing block vectors;

g) a default block vector (related to a block size).

[0369] After adding the block vector for the neighbouring block / the block vector for the non-neighbouring block, a process of AR-BVP construction is implemented. Then, after adding the block vector in the history-based block vector buffer list, a process of AR-BVP construction is implemented.

[0370] 3. A process of AR-BVP construction is determined. Guiding block vectors for AR-BVP construction can include: all block vectors already in the list, some (one or more) block vectors in the list, and a newly constructed block vector (AR-BVP). After a certain guiding block vector is determined, one or more AR-BVPs can be constructed. A termination condition for the construction process can include: the number of iterations of AR-BVP

construction, the number of AR-BVPs already added, and the current number of the list. Processes of different AR-BVP construction can be different. For example, during the first AR-BVP construction, all block vectors already in the list are used as guiding block vectors, and construction is not continued based on AR-BVPs already added. During the second AR-BVP construction, all block vectors already in the list and a newly added AR-BVP are used as guiding block vectors, and each guiding block vector is used to construct multiple AR-BVPs.

[0371] 4. For a given guiding block vector BV_0 and a block vector BV_ref for a reference block, besides summing BV_0 and BV_ref to obtain BV_arbvp = BV_0 + BV_ref, BV_ref can also be directly used as a candidate block vector, that is, BV_ext = BV_ref, which is then checked and added to the block vector list. For example, when constructing AR-BVP, after obtaining a block vector for a reference block according to a guiding block vector each time, each of BV_arbvp and BV_ext can be checked and added.

[0372] 5. If an IBC merge mode is to be applied to the current block, a merge index value is determined, a corresponding block vector is selected from the block vector list, and a reference block is obtained as a prediction block for the current block.

[0373] 6. If an IBC AMVP mode is to be applied to the current block, a block vector for the current block is determined through search at an encoding end, and a reference block is obtained as a prediction block for the current block. A BVP index value is determined, and a corresponding block vector is selected from the block vector list. A BVD is determined according to the selected block vector and the block vector for the current block.

[0374] 7. A residual block is obtained according to the prediction block for the current block and original values of the current block. The residual block is transformed and quantized, and is then encoded through entropy coding. In addition, coding information such as cu_pred_mode, the merge index value, and the BVD is encoded, thereby completing encoding of the current block.

[0375] It should be noted that, experimental test is conducted on the encoding method proposed in embodiments of the disclosure. When multi-time AR-BVP construction is

applied in the IBC mode, a Y-component coding gain of approximately -0.01% can be achieved on an ECM reference software test platform.

[0376] It should be noted that, experimental test is conducted on the encoding method proposed in embodiments of the disclosure. When a block vector for a reference block is directly used as a candidate block vector in the IntraTMP mode, a Y-component coding gain of approximately -0.03% can be achieved.

[0377] The encoding method proposed in embodiments of the disclosure is exemplarily described below by taking the DIMD mode and the first syntax flag information being cu_dimd_flag as an example. In the DIMD mode, a block vector list of subsample precision can be constructed, and a subsample interpolation-based prediction block is obtained according to block vectors in the block vector list and then fused with prediction blocks obtained through intra prediction to obtain a final prediction block. For example, the subsample interpolation-based prediction block obtained is fused with prediction blocks generated in intra modes derived from HoG.

[0378] Encoding process:

[0379] 1. A current block is encoded, and it is determined to select the DIMD prediction mode for encoding the current block.

[0380] 2. A HoG is obtained according to a reconstructed region (or encoded information), and thus intra angular modes are derived, so as to generate corresponding prediction blocks.

[0381] 3. A block vector list is constructed, where the precision of candidate block vectors is subsample precision. For example, the precision of a block vector is 1/16 samples. The candidate block vectors can be obtained from a neighbouring block, a non-neighbouring block, an AR-BVP candidate, a history-based block vector buffer, etc.

[0382] 4. Template error values for *N* block vectors in the block vector list are calculated. For example, the width of a template is Tw, and the height of the template Th. The width of the current block is W, and the height of the current block is H. A template region is illustrated in FIG. 15.

[0383] The candidate block vector is BV, and prediction on a top template and prediction on a left template are derived. For example, the top template region is obtained. Adjustment is performed on BV according to the height Th of the template, BV_top = BV + (Th << mv_bits), where mv_bits is a subsample bit number. Taking a top-left corner of the current block as a reference, a reference block with a width of W and a height of Th is obtained according to BV_top. If the reference block is unavailable, BV_top is adjusted to be equal to BV and a reference block is obtained. If BV includes a subsample part, a reference block is obtained based on subsample interpolation. An interpolation filter can be a 2-tap interpolation filter, which can be the same as a 2-tap interpolation filter used in inter prediction or IBC prediction. The obtained reference block is used as the prediction on the top template region. The prediction on the left template region is derived in a similar way. Template error values, such as SAD values, are calculated according to prediction values for a template region and reconstructed values for the template region.

[0384] 5. A template error value in a predefined intra non-angular mode can be calculated. For example, one mode Planar is predefined, and a template error value in the Planar mode is calculated according to reference samples.

[0385] 6. An optimal non-angular prediction method is selected according to the candidate block vectors and the template error value in the intra non-angular mode. If the optimal prediction method is to use a certain candidate block vector, in a case that the block vector is of subsample precision, a reference block for the current block can be obtained by means of interpolation filtering, where an interpolation filter can adopt an *n*-tap filter, for example, $n = 8$, which can be the same as an interpolation filter used in IBC prediction.

[0386] 7. According to the above steps, the optimal non-angular prediction method is obtained, and a prediction block generated in the optimal non-angular prediction method is fused with prediction blocks generated in intra angular modes derived from HoG, to obtain a final prediction block. A residual block is obtained according to the prediction block for the current block and original values of the current block. The residual block is transformed and quantized, and is then encoded through entropy coding. In addition, coding information such as cu_dimd_flag is encoded, thereby completing encoding of the current block.

[0387] It should be noted that, an experimental test is conducted on the encoding method proposed in embodiments of the disclosure. When a block vector list of subsample-precision is applied in the DIMD mode or the SGPM mode, a Y-component coding gain of approximately -0.01% can be achieved.

[0388] The encoding method proposed in embodiments of the disclosure is exemplarily described below by taking the IntraTMP mode, the first syntax flag information being cu_tmp_flag, the second syntax flag information being cu_tmp_lic_flag, and the third syntax flag information being cu_tmp_idx as an example. A block vector list can be reordered according to template errors. Template region selection and template-error calculation can be implemented in different methods. For example, the template region and the error calculation method for the block vector list can be predefined, or can be consistent with a current prediction process. Taking the IntraTMP mode as an example, for the current block, error calculation methods such as SAD and MR-SAD can be used, and a template region can be classified into an L-shaped template, a top template, and a left template. Corresponding syntax elements in a block vector list-based prediction process are as follows:

...

cu_tmp_flag

if(cu_tmp_flag)

{

...

cu_tmp_lic_flag

...

cu_tmp_idx

...

}

...

[0389] Encoding process:

[0390] 1. A current block is encoded, and it is determined to select the IntraTMP prediction mode for the current block.

[0391] 2. A block vector list is constructed. Candidate block vectors can be obtained from a neighbouring block, a non-neighbouring block, an AR-BVP candidate, a history-based block vector buffer, etc.

[0392] 3. Template error values for candidate block vectors in the block vector list are calculated, and the block vector list is reordered based on the template error values and the first *M* candidates are retained. Template error values corresponding to each of an L-shaped template, a top template, and a left template under each of SAD and MR-SAD can be calculated respectively, to obtain corresponding sorted block vector lists, which are denoted as mrgList0, mrgList1, mrgList2, mrgListMR0, mrgListMR1, and mrgListMR2.

[0393] 4. A maximum template error value pDiff can be obtained according to step 3, which is related to the template error values for the candidates in the block vector list. For example, pDiff can be equal to the maximum of diff among the candidates in mrgList plus 1. During subsequent sparse search and refinement search, template error values for candidate block vectors found need to be less than pDiff. Corresponding pDiff can be set with regard to different template-error calculation methods and template types.

[0394] 5. Sparse search is performed to obtain a search result corresponding to each of the L-shaped template, the top template, and the left template under each of SAD and MR-SAD, which can be denoted as sparseList0, sparseList1, sparseList2, sparseListMR0, sparseListMR1, and sparseListMR2. Corresponding block vector lists are merged with the sparse search results. Refinement search is performed according to the merged lists, to obtain a final IntraTMP block vector candidate list under SAD and a final IntraTMP block vector candidate list under MR-SAD.

[0395] 6. Whether to apply the IntraTMP block vector candidate list corresponding to SAD or to apply the IntraTMP block vector candidate list corresponding to MR-SAD to the current block is determined, and a corresponding value of cu_tmp_lic_flag is set. Then, a block vector selected for the current block from the list is determined, which is represented by an index value cu_tmp_idx. A reference block is obtained according to the block vector

and used as a prediction block for the current block. A residual block is obtained according to the prediction block for the current block and original values of the current block. The residual block is transformed and quantized, and is then encoded through entropy coding. In addition, coding information such as cu_tmp_flag, cu_tmp_idx, and cu_tmp_lic_flag is encoded, thereby completing encoding of the current block.

[0396] It should be noted that, in embodiments of the disclosure, in the IntraTMP mode, SATD and MR-SATD can be used for template-error calculation. In an implementation, SAD is used in a sparse search stage and SATD is used in a refinement search stage. If cu_tmp_lic_flag for the current block is true, MR-SAD and MR-SATD are used respectively. SATD or MR-SATD can be used when calculating template error values for the block vector list, which is consistent with the refinement search stage. In another implementation, whether SATD is to be used for search is indicated via a flag cu_tmp_satd_flag signalled into a bitstream. For example, SAD is to be used when cu_tmp_satd_flag is false, and SATD is to be used when cu_tmp_satd_flag is true. A method for calculating template error values for the block vector list can be determined according to cu_tmp_satd_flag. SAD is to be used when cu_tmp_satd_flag is false, and SATD is to be used when cu_tmp_satd_flag is true. Alternatively, both cu_tmp_satd_flag and cu_tmp_lic_flag can be encoded to determine which one of SAD, SATD, MR-SAD, or MR-SATD is to be used for search and calculation of template errors for the block vector list.

[0397] It should be noted that, in embodiments of the disclosure, a candidate list sparseList obtained through sparse search can be used as guiding block vectors to construct AR-BVP candidates, and template error values for these AR-BVP candidates are calculated and sparseList is updated (*n* candidates with the smallest template error values are retained in the list). AR-BVP candidate construction can be performed for sparseList corresponding to different template types, and sparseList is updated according to template error values for AR-BVP candidates.

[0398] It should be noted that, in embodiments of the disclosure, when the block vector list is used for a search process in the IntraTMP mode, the block vector list can be used only for a search corresponding to one template type. For example, in Embodiment 3, a block vector list mrgList is constructed and reordered according to an L-shaped template, and a

candidate list sparseList0 corresponding to the L-shaped template, a candidate list sparseList1 corresponding to the top template, and a candidate list sparseList2 corresponding to the left template are obtained through sparse search. mrgList can be merged with only sparseList0 (the candidate list corresponding to the L-shaped template).

[0399] It should be noted that, an experimental test is conducted on the encoding method proposed in embodiments of the disclosure. When the block vector list is reordered by using SAD and MR-SAD respectively in the IntraTMP mode, a Y-component coding gain of approximately -0.02% can be achieved.

[0400] Embodiments of the disclosure provide a coding method. If the preset intra prediction mode is to be applied to the current block, the block vector list for the current block is constructed; and the prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes the block vector for the reference block corresponding to the guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in the preset template-error calculation method, and the preset template-error calculation method is determined according to the error calculation method for the template search process corresponding to the current block or the second syntax flag information transmitted in the bitstream. That is, in embodiments of the disclosure, the construction and/or usage process for the block vector list during intra prediction can be adjusted and improved. By constructing AR-BVP candidate block vectors multiple times, and/or directly using a block vector for a reference block as a candidate block vector to be added to the block vector list, and/or using a subsample-precision block vector list for intra prediction, and/or reordering the block vector list in different template-error calculation methods, diversity of the block vector list can be improved, thereby improving intra prediction efficiency and improving coding performance.

[0401] Based on the above embodiments, in another embodiment of the disclosure, with the same inventive concept as the above embodiments, FIG. 17 is a schematic structural diagram of an encoder provided in embodiments of the disclosure. As illustrated in FIG. 17, the encoder 170 can include a first determining unit 1701. The first determining unit 1701 is configured to: signal first syntax flag information into a bitstream when it is determined that a preset intra prediction mode is to be applied to a current block, where the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block; construct a block vector list for the current block; and determine a prediction block for the current block according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0402] It can be understood that, in embodiments of the disclosure, the term “part” can be a portion of circuit, a portion of a processor, a portion of a program or software, or can be modular or non-modular. In addition, various components described in embodiments of the disclosure may be integrated into one processing unit or may be present as a number of physically separated units, or two or more units may be integrated into one. The integrated unit may take the form of hardware or a software functional unit.

[0403] If the integrated unit is implemented as software functional modules and not sold or used as standalone products, it may be stored in a computer-readable storage medium. Based on such an understanding, the essential technical solutions of embodiments,

or the portion that contributes to the related art, or all or part of the technical solutions may be embodied as software products. The computer software products can be stored in a storage medium and may include multiple instructions that, when executed, can cause a computer device, e.g., a personal computer, a server, a network device, a processor, etc., to execute some or all operations of the methods described in various embodiments of the disclosure. The above storage medium may include various kinds of media that can store program codes, such as a universal serial bus (USB) flash disk, a mobile hard drive, a read-only memory (ROM), a random access memory (RAM), a magnetic disk, an optical disk, etc.

[0404] Therefore, embodiments of the disclosure provide a computer-readable storage medium. The computer-readable storage medium is applied to the encoder 170. The computer-readable storage medium stores computer programs which, when executed by a first processor, implement the method according to any of the foregoing embodiments.

[0405] Based on the composition of the encoder 170 and the computer-readable storage medium described above, referring to FIG. 18, FIG. 18 is a schematic hardware structural diagram of an encoder provided in embodiments of the disclosure. As illustrated in FIG. 18, the encoder 170 may include: a first communication interface 1801, a first memory 1802, and a first processor 1803. These components are coupled together via a first bus system 1804. It can be understood that, the first bus system 1804 is configured for connection and communication between these components. In addition to a data bus, the first bus system 1804 further includes a power bus, a control bus, and a status signal bus. However, for clarity of illustration, all buses are labeled as the first bus system 1804 in FIG. 18. The first communication interface 1801 is configured for signal reception and transmission during information reception and transmission with other external network elements. The first memory 1802 is configured to store computer programs executable by the first processor 1803. The first processor 1803 is configured to, when running the computer programs, perform the following operations: signal first syntax flag information into a bitstream when it is determined that a preset intra prediction mode is to be applied to a current block, where the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block; construct a block vector list for the current block; and determine a prediction block for the current block according to the block vector list for

the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0406] It can be understood that, the first memory 1802 in embodiments of the disclosure may be a volatile memory or a non-volatile memory, or may include both the volatile memory and the non-volatile memory. The non-volatile memory may be a ROM, a programmable ROM (PROM), an erasable PROM (EPROM), an electrically EPROM (EEPROM), or flash memory. The volatile memory can be a RAM that acts as an external cache. By way of example but not limitation, many forms of RAM are available, such as a static RAM (SRAM), a dynamic RAM (DRAM), a synchronous DRAM (SDRAM), a double data rate SDRAM (DDR SDRAM), an enhanced SDRAM (ESDRAM), a synchlink DRAM (SLDRAM), and a direct rambus RAM (DR RAM). It should be noted that, the first memory 1802 of the systems and methods described in the disclosure is intended to include, but is not limited to, these and any other suitable types of memory.

[0407] The first processor 1803 in embodiments of the disclosure may be an integrated circuit chip with signal processing capabilities. During implementation, each step of the foregoing method embodiments may be completed by an integrated logic circuit of hardware in the first processor 1803 or an instruction in the form of software. The processor may be a general-purpose processor, a digital signal processor (DSP), an application specific integrated circuit (ASIC), a field programmable gate array (FPGA), or other programmable logic devices, discrete gates or transistor logic devices, discrete hardware components. The

methods, steps, and logic blocks disclosed in embodiments of the disclosure can be implemented or executed. The general-purpose processor may be a microprocessor, or the processor may be any conventional processor or the like. The steps of the method disclosed in embodiments of the disclosure may be directly implemented by a hardware decoding processor, or may be performed by hardware and software modules in the decoding processor. The software module can be located in a storage medium such as a RAM, a flash memory, a ROM, a PROM, or an electrically erasable programmable memory, registers, and the like. The storage medium is located in the first memory 1802. The first processor 1803 reads the information in the first memory 1802, and completes the steps of the method described above with the hardware thereof.

[0408] It is understandable that the embodiments described in the disclosure may be implemented in hardware, software, firmware, middleware, microcode, or any combination thereof. For a hardware implementation, the processing unit may be implemented within one or more ASICs, DSPs, DSP devices (DSPDs), PLDs, FPGAs, general-purpose processors, controllers, microcontrollers, microprocessors, other electronic units designed to perform the functions described in the disclosure, or a combination thereof. For a software implementation, the techniques described in the disclosure may be realized through modules (e.g., procedures, functions, etc.) for performing the functions described in the disclosure. Software codes may be stored in a memory and executed by a processor. The memory may be implemented within the processor or externally to the processor.

[0409] Optionally, as another embodiment, the first processor 1803 is further configured to perform the method according to any of the foregoing embodiments when running the computer programs.

[0410] In another embodiment of the disclosure, with the same inventive concept as the above embodiments, FIG. 19 is a schematic structural diagram of a decoder provided in embodiments of the disclosure. As illustrated in FIG. 19, the decoder 190 can include a second determining unit 1901. The second determining unit 1901 is configured to: decode a bitstream to determine first syntax flag information; when the first syntax flag information indicates that a preset intra prediction mode is to be applied to a current block, construct a block vector list for the current block; and determine a prediction block for the current block

according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0411] It can be understood that, in the embodiments, the term “part” can be a portion of circuit, a portion of a processor, a portion of a program or software, or can be modular or non-modular. In addition, various components described in various embodiments of the disclosure may be integrated into one processing unit or may be present as a number of physically separated units, or two or more units may be integrated into one. The integrated unit may take the form of hardware or a software functional unit.

[0412] If the integrated units are implemented in the form of software functional modules and are not sold or used as standalone products, they may be stored in a computer-readable storage medium. Based on such understanding, embodiments provide a computer-readable storage medium. The computer-readable storage medium is applied to a decoder 190. The computer-readable storage medium stores computer programs which, when executed by a second processor, implement the method according to any of the foregoing embodiments.

[0413] Based on the composition of the decoder 190 and the computer-readable storage medium, referring to FIG. 20, FIG. 20 is a schematic hardware structural diagram of a decoder provided in embodiments of the disclosure. As illustrated in FIG. 20, the decoder 190 may include: a second communication interface 2001, a second memory 2002, and a

second processor 2003. These components are coupled together via a second bus system 2004. It can be understood that, the second bus system 2004 is configured for connection and communication between these components. In addition to a data bus, the second bus system 2004 further includes a power bus, a control bus, and a status signal bus. However, for clarity of illustration, all buses are labeled as the second bus system 2004 in FIG. 20. The second communication interface 2001 is configured for signal reception and transmission during information reception and transmission with other external network elements. The second memory 2002 is configured to store computer programs executable by the second processor 2003. The second processor 2003 is configured to perform the following operations when running the computer programs: decode a bitstream to determine first syntax flag information; when the first syntax flag information indicates that a preset intra prediction mode is to be applied to a current block, construct a block vector list for the current block; and determine a prediction block for the current block according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an AR-BVP process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for DIMD, or TIMD, or SGPM. The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

[0414] Optionally, as another embodiment, the second processor 2003 is further configured to perform the method according to any of the foregoing embodiments when running the computer programs.

[0415] It is understandable that, the hardware functionality of the second memory 2002 is similar to that of the first memory 1802, and the hardware functionality of the second

processor 2003 is similar to that of the first processor 1803, which will not be elaborated in detail herein.

[0416] In another embodiment of the disclosure, referring to FIG. 21, which is a schematic structural diagram of a coding system provided in embodiments of the disclosure. As illustrated in FIG. 21, the coding system 210 may include an encoder 2101 and a decoder 2102.

[0417] In embodiments of the disclosure, the encoder 2101 can be the encoder described in any one of the foregoing embodiments, and the decoder 2102 can be the decoder described in any one of the foregoing embodiments.

[0418] It should be noted that, in the disclosure, the terms “include”, “comprise”, or any other variants thereof are intended to cover non-exclusive inclusion. This means that a process, method, article, or apparatus that includes a series of elements not only includes those elements but may further include other elements not explicitly listed, or further include elements inherent to such process, method, article, or apparatus. In the absence of more limitations, an element defined by the phrase “include/comprise a...” does not exclude the existence of additional identical elements in the process, method, article, or apparatus that includes said element.

[0419] The serial numbers in embodiments of the disclosure are merely for illustrative purposes and do not represent the superiority or inferiority of the embodiments.

[0420] The methods disclosed in the several method embodiments provided in the disclosure may be arbitrarily combined without conflict to obtain new method embodiments.

[0421] The features disclosed in the several product embodiments provided in the disclosure may be arbitrarily combined without conflict to obtain new product embodiments.

[0422] The features disclosed in the several method or device embodiments provided in the disclosure may be arbitrarily combined without conflict to obtain new method embodiments or device embodiments.

[0423] The foregoing elaborations are merely implementations of the disclosure, but are not intended to limit the protection scope of the disclosure. Any variation or replacement

easily thought of by those skilled in the art within the technical scope disclosed in the disclosure shall belong to the protection scope of the disclosure. Therefore, the protection scope of the disclosure shall be subject to the protection scope of embodiments of the disclosure.

INDUSTRIAL APPLICABILITY

[0424] Embodiments of the disclosure provide a coding method, a bitstream, an encoder, a decoder, and a storage medium. If a preset intra prediction mode is to be applied to a current block, a block vector list for the current block is constructed; and a prediction block for the current block is determined according to the block vector list for the current block. The block vector list for the current block satisfies at least one of the following. The block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector. The block vector list includes a block vector for a reference block corresponding to a guiding block vector for the current block. The block vector list includes a subsample-precision block vector, where the subsample-precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM). The block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list, where the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in a bitstream. That is, in embodiments of the disclosure, the construction and/or usage process for the block vector list during intra prediction can be adjusted and improved. By constructing AR-BVP candidate block vectors multiple times, and/or directly using a block vector for a reference block as a candidate block vector to be added to the block vector list, and/or using a subsample-precision block vector list for intra prediction, and/or reordering the block vector list in different template-error calculation methods, diversity of the block vector list can be improved, thereby improving intra prediction efficiency and improving coding performance.

CLAIMS

I/We claim:

1. A decoding method, applied to a decoder and comprising:

parsing a bitstream to determine first syntax flag information;

constructing a block vector list for a current block when the first syntax flag information indicates to apply a preset intra prediction mode to the current block; and

determining a prediction block for the current block according to the block vector list for the current block;

wherein the block vector list for the current block satisfies at least one of:

the block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector;

the block vector list comprises a block vector for a reference block corresponding to a guiding block vector for the current block;

the block vector list comprises a subsample-precision block vector; wherein the subsample-precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM); or

the block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list; wherein the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

2. The method of claim 1, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first candidate block vector for the current block according to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in a history-based motion vector prediction (HMVP) buffer;

adding the first candidate block vector and a second block vector to the block vector list; wherein the second block vector is a non-AR-BVP block vector;

determining a third block vector in the block vector list as a second guiding block vector for the current block, and determining a second candidate block vector for the current block according to the second guiding block vector; wherein the third block vector at least comprises the first candidate block vector; and

adding the second candidate block vector to the block vector list.

3. The method of claim 1, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first reference block corresponding to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer;

determining a block vector for the first reference block as a first candidate block vector for the current block; and/or summing the first guiding block vector and the block vector for the first reference block to determine the first candidate block vector for the current block; and

adding the first candidate block vector to the block vector list.

4. The method of claim 1, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first reference block

corresponding to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer;

determining a block vector for the first reference block as a first candidate block vector for the current block; and/or summing the first guiding block vector and the block vector for the first reference block to determine the first candidate block vector for the current block;

adding the first candidate block vector and a second block vector to the block vector list; wherein the second block vector is a non-AR-BVP block vector;

determining a third block vector in the block vector list as a second guiding block vector for the current block, and determining a second reference block corresponding to the second guiding block vector; wherein the third block vector at least comprises the first candidate block vector;

determining a block vector for the second reference block as a second candidate block vector for the current block; and/or summing the second guiding block vector and the block vector for the second reference block to determine the second candidate block vector for the current block; and

adding the second candidate block vector to the block vector list.

5. The method of claim 4, further comprising:

performing an AR-BVP process according to the guiding block vector for the current block to obtain at least one candidate block vector, to construct the block vector list for the current block, until a preset termination condition is satisfied; wherein the preset termination condition comprises at least one of:

the number of AR-BVP processes is greater than or equal to a preset threshold;

the number of candidate block vectors determined through AR-BVP processes is greater than or equal to a first number threshold; or

the number of block vectors in the block vector list is greater than or equal to a second number threshold.

6. The method of claim 2 or 3, further comprising:

determining, according to the number of block vectors in the block vector list, the number of block vectors in the HMVP buffer to be added to the block vector list.

7. The method of claim 4, wherein determining the prediction block for the current block according to the block vector list for the current block comprises:

determining template error values for block vectors in the block vector list according to a subsample-precision block vector in the block vector list; and

determining the prediction block for the current block according to the template error values for the block vectors in the block vector list.

8. The method of claim 7, wherein determining the template error values for the block vectors in the block vector list according to the subsample-precision block vector in the block vector list comprises:

determining a template type parameter and a template size parameter corresponding to the template type parameter;

determining an adjusted block vector corresponding to the subsample-precision block vector according to the template type parameter, the template size parameter, and a subsample bit parameter for the subsample-precision block vector;

determining prediction values for a reference template according to the adjusted block vector; and

determining the template error values for the block vectors in the block vector list according to the prediction values for the reference template and reconstructed values for a current template.

9. The method of claim 8, wherein determining the prediction values for the reference template according to the adjusted block vector comprises:

when a candidate reference template corresponding to the adjusted block vector is available, determining the prediction values for the reference template according to the adjusted block vector and a subsample interpolation algorithm;

when the candidate reference template corresponding to the adjusted block vector is unavailable, determining the prediction values for the reference template according to block vectors in the block vector list and the subsample interpolation algorithm.

10. The method of any of claims 4 and 7 to 9, wherein determining the prediction block for the current block according to the block vector list for the current block comprises:

performing block-vector template-error calculation on block vectors in the block vector list to determine template error values for the block vectors in the block vector list;

sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list; and

determining the prediction block for the current block based on the sorted block vector list.

11. The method of claim 10, wherein performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list comprises:

parsing the bitstream to determine the second syntax flag information;

determining a preset template-error calculation method according to the second syntax flag information; wherein the preset template-error calculation method comprises any one of: sum of absolute differences (SAD), mean-removed SAD (MR-SAD), sum of absolute transformed differences (SATD), or MR-SATD; and

determining template error values for the block vectors in the block vector list corresponding to at least one template type based on the preset template-error calculation method.

12. The method of claim 11, wherein determining the preset template-error calculation method according to the second syntax flag information comprises:

determining a first template-error calculation method and a second template-error calculation method according to the second syntax flag information; wherein the first template-error calculation method is used for performing sparse search on the at least one template type, and the second template-error calculation method is used for performing

refinement search on the at least one template type and block-vector template-error calculation.

13. The method of claim 11, wherein determining the preset template-error calculation method according to the second syntax flag information comprises:

determining a third template-error calculation method and a fourth template-error calculation method according to the second syntax flag information; wherein the third template-error calculation method is used for performing sparse search on the at least one template type and block-vector template-error calculation, and the fourth template-error calculation method is used for performing refinement search on the at least one template type.

14. The method of claim 11, wherein determining the preset template-error calculation method according to the second syntax flag information comprises:

determining a fifth template-error calculation method according to the second syntax flag information; wherein the fifth template-error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

15. The method of claim 11, wherein the second syntax flag information comprises first information and second information, and determining the preset template-error calculation method according to the second syntax flag information comprises:

determining a sixth error calculation method according to the first information and the second information; wherein the sixth error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

16. The method of claim 10, wherein performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list comprises:

performing block-vector template-error calculation on the block vectors in the block vector list according to a preset template-error calculation method for sparse search in a template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list; or

performing block-vector template-error calculation on the block vectors in the block vector list according to a preset template-error calculation method for refinement search in the template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list.

17. The method of any of claims 10 to 15, wherein sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain the sorted block vector list comprises:

parsing the bitstream to determine third syntax flag information;

determining at least one template-number corresponding to the at least one template type according to the third syntax flag information; and

sorting the template error values for the block vectors in the block vector list corresponding to the at least one template type separately according to the at least one template-number, to obtain at least one first candidate list corresponding to the at least one template type.

18. The method of claim 17, wherein determining the prediction block for the current block based on the sorted block vector list comprises:

performing sparse search separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively;

merging the at least one first candidate list and the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type;

performing refinement search separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; and

determining a target candidate list corresponding to the current block according to the at least one fourth candidate list, and determining the prediction block for the current block according to the target candidate list.

19. The method of claim 18, further comprising:

for any one of the second candidate lists, determining a third guiding block vector according to the second candidate list, and determining a third candidate block vector for the current block according to the third guiding block vector; and

determining a template error value corresponding to the third candidate block vector, and updating the second candidate list according to the template error value corresponding to the third candidate block vector.

20. The method of claim 18, wherein determining the prediction block for the current block based on the sorted block vector list comprises:

performing sparse search separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively;

merging a first candidate list corresponding to a preset template type and the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type;

performing refinement search separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; and

determining a target candidate list corresponding to the current block according to the at least one fourth candidate list, and determining the prediction block for the current block according to the target candidate list.

21. The method of any of claims 18 to 20, wherein

during refinement search based on the at least one third candidate list, all block vectors in the third candidate list correspond to the same template search region; or

during refinement search based on the at least one third candidate list, fourth block vectors in the third candidate list correspond to the same template search region; wherein the fourth block vectors are block vectors derived from the first candidate list.

22. An encoding method, applied to an encoder and comprising:

signalling first syntax flag information into a bitstream when it is determined that a preset intra prediction mode is to be applied to a current block; wherein the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block;

constructing a block vector list for the current block; and

determining a prediction block for the current block according to the block vector list for the current block;

wherein the block vector list for the current block satisfies at least one of:

the block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector;

the block vector list comprises a block vector for a reference block corresponding to a guiding block vector for the current block;

the block vector list comprises a subsample-precision block vector; wherein the subsample-precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM); or

the block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list; wherein the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

23. The method of claim 22, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first candidate block vector for the current block according to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in a history-based motion vector prediction (HMVP) buffer;

adding the first candidate block vector and a second block vector to the block vector list; wherein the second block vector is a non-AR-BVP block vector;

determining a third block vector in the block vector list as a second guiding block vector for the current block, and determining a second candidate block vector for the current block according to the second guiding block vector; wherein the third block vector at least comprises the first candidate block vector; and

adding the second candidate block vector to the block vector list.

24. The method of claim 22, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first reference block corresponding to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer;

determining a block vector for the first reference block as a first candidate block vector for the current block; and/or summing the first guiding block vector and the block vector for the first reference block to determine the first candidate block vector for the current block; and

adding the first candidate block vector to the block vector list.

25. The method of any of claim 22, wherein constructing the block vector list for the current block comprises:

determining a first block vector in the block vector list for the current block as a first guiding block vector for the current block, and determining a first reference block

corresponding to the first guiding block vector; wherein the first block vector comprises at least one of: a block vector for a neighbouring block of the current block, a block vector for a non-neighbouring block of the current block, or a block vector in an HMVP buffer;

determining a block vector for the first reference block as a first candidate block vector for the current block; and/or summing the first guiding block vector and the block vector for the first reference block to determine the first candidate block vector for the current block;

adding the first candidate block vector and a second block vector to the block vector list; wherein the second block vector is a non-AR-BVP block vector;

determining a third block vector in the block vector list as a second guiding block vector for the current block, and determining a second reference block corresponding to the second guiding block vector; wherein the third block vector at least comprises the first candidate block vector;

determining a block vector for the second reference block as a second candidate block vector for the current block; and/or summing the second guiding block vector and the block vector for the second reference block to determine the second candidate block vector for the current block; and

adding the second candidate block vector to the block vector list.

26. The method of claim 25, further comprising:

performing an AR-BVP process according to the guiding block vector for the current block to obtain at least one candidate block vector, to construct the block vector list for the current block, until a preset termination condition is satisfied; wherein the preset termination condition comprises at least one of:

the number of AR-BVP processes is greater than or equal to a preset threshold;

the number of candidate block vectors determined through AR-BVP processes is greater than or equal to a first number threshold; or

the number of block vectors in the block vector list is greater than or equal to a second number threshold.

27. The method of claim 23 or 24, further comprising:

determining, according to the number of block vectors in the block vector list, the number of block vectors in the HMVP buffer to be added to the block vector list.

28. The method of claim 25, wherein determining the prediction block for the current block according to the block vector list for the current block comprises:

determining template error values for block vectors in the block vector list according to a subsample-precision block vector in the block vector list; and

determining the prediction block for the current block according to the template error values for the block vectors in the block vector list.

29. The method of claim 28, wherein determining the template error values for the block vectors in the block vector list according to the subsample-precision block vector in the block vector list comprises:

determining a template type parameter and a template size parameter corresponding to the template type parameter;

determining an adjusted block vector corresponding to the subsample-precision block vector according to the template type parameter, the template size parameter, and a subsample bit parameter for the subsample-precision block vector;

determining prediction values for a reference template according to the adjusted block vector; and

determining the template error values for the block vectors in the block vector list according to the prediction values for the reference template and reconstructed values for a current template.

30. The method of claim 29, wherein determining the prediction values for the reference template according to the adjusted block vector comprises:

when a candidate reference template corresponding to the adjusted block vector is available, determining the prediction values for the reference template according to the adjusted block vector and a subsample interpolation algorithm;

when the candidate reference template corresponding to the adjusted block vector is unavailable, determining the prediction values for the reference template according to block vectors in the block vector list and the subsample interpolation algorithm.

31. The method of any of claims 25 and 28 to 30, wherein determining the prediction block for the current block according to the block vector list for the current block comprises:

performing block-vector template-error calculation on block vectors in the block vector list to determine template error values for the block vectors in the block vector list;

sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain a sorted block vector list; and

determining the prediction block for the current block based on the sorted block vector list.

32. The method of claim 31, wherein performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list comprises:

determining a preset template-error calculation method, and signalling the second syntax flag information into the bitstream; wherein the second syntax flag information is used to determine the preset template-error calculation method; and the preset template-error calculation method comprises any one of: sum of absolute differences (SAD), mean-removed SAD (MR-SAD), sum of absolute transformed differences (SATD), or MR-SATD; and

determining template error values for the block vectors in the block vector list corresponding to at least one template type based on the preset template-error calculation method.

33. The method of claim 32, wherein determining the preset template-error calculation method comprises:

determining a first template-error calculation method and a second template-error calculation method; wherein the first template-error calculation method is used for

performing sparse search on the at least one template type, and the second template-error calculation method is used for performing refinement search on the at least one template type and block-vector template-error calculation.

34. The method of claim 32, wherein determining the preset template-error calculation method comprises:

determining a third template-error calculation method and a fourth template-error calculation method; wherein the third template-error calculation method is used for performing sparse search on the at least one template type and block-vector template-error calculation, and the fourth template-error calculation method is used for performing refinement search on the at least one template type.

35. The method of claim 32, wherein determining the preset template-error calculation method comprises:

determining a fifth template-error calculation method; wherein the fifth template-error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation.

36. The method of claim 32, wherein the second syntax flag information comprises first information and second information, and determining the preset template-error calculation method comprises:

determining a sixth error calculation method; wherein the sixth error calculation method is used for performing sparse search on the at least one template type, refinement search on the at least one template type, and block-vector template-error calculation; and

determining the first information and the second information according to the sixth error calculation method.

37. The method of claim 31, wherein performing block-vector template-error calculation on the block vectors in the block vector list to determine the template error values for the block vectors in the block vector list comprises:

performing block-vector template-error calculation on the block vectors in the block vector list according to a preset template-error calculation method for sparse search in a template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list; or

performing block-vector template-error calculation on the block vectors in the block vector list according to a preset template-error calculation method for refinement search in the template search process corresponding to the current block, to determine the template error values for the block vectors in the block vector list.

38. The method of any of claims 31 to 36, wherein sorting the block vector list according to the template error values for the block vectors in the block vector list to obtain the sorted block vector list comprises:

determining at least one template-number corresponding to the at least one template type, and signalling third syntax flag information into the bitstream; wherein the third syntax flag information is used to determine the at least one template-number corresponding to the at least one template type; and

sorting the template error values for the block vectors in the block vector list corresponding to the at least one template type separately according to the at least one template-number, to obtain at least one first candidate list corresponding to the at least one template type.

39. The method of claim 38, wherein determining the prediction block for the current block based on the sorted block vector list comprises:

performing sparse search separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively;

merging the at least one first candidate list and the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type;

performing refinement search separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; and

determining a target candidate list corresponding to the current block according to the at least one fourth candidate list, and determining the prediction block for the current block according to the target candidate list.

40. The method of claim 39, further comprising:

for any one of the second candidate lists, determining a third guiding block vector according to the second candidate list, and determining a third candidate block vector for the current block according to the third guiding block vector; and

determining a template error value corresponding to the third candidate block vector, and updating the second candidate list according to the template error value corresponding to the third candidate block vector.

41. The method of claim 39, wherein determining the prediction block for the current block based on the sorted block vector list comprises:

performing sparse search separately for the at least one template type, to obtain at least one second candidate list corresponding to the at least one template type respectively;

merging a first candidate list corresponding to a preset template type and the at least one second candidate list respectively to obtain at least one third candidate list corresponding to the at least one template type;

performing refinement search separately for the at least one template type based on the at least one third candidate list, to obtain at least one fourth candidate list corresponding to the at least one template type respectively; and

determining a target candidate list corresponding to the current block according to the at least one fourth candidate list, and determining the prediction block for the current block according to the target candidate list.

42. The method of any of claims 39 to 41, wherein

during refinement search based on the at least one third candidate list, all block vectors in the third candidate list correspond to the same template search region; or

during refinement search based on the at least one third candidate list, fourth block vectors in the third candidate list correspond to the same template search region; wherein the fourth block vectors are block vectors derived from the first candidate list.

43. A bitstream, generated by encoding bits according to information to be encoded; wherein the information to be encoded comprises at least one of: first syntax flag information, second syntax flag information, or third syntax flag information.

44. An encoder, comprising:

a first determining unit configured to:

signal first syntax flag information into a bitstream when it is determined that a preset intra prediction mode is to be applied to a current block; wherein the first syntax flag information indicates that the preset intra prediction mode is to be applied to the current block;

construct a block vector list for the current block; and

determine a prediction block for the current block according to the block vector list for the current block; wherein the block vector list for the current block satisfies at least one of:

the block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector;

the block vector list comprises a block vector for a reference block corresponding to a guiding block vector for the current block;

the block vector list comprises a subsample-precision block vector; wherein the subsample-precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM); or

the block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list; wherein the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined

according to an error calculation method for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

45. An encoder, comprising a first memory and a first processor, wherein:

the first memory is configured to store computer programs executable by the first processor; and

the first processor is configured to, when executing the computer programs, perform the method of any of claims 22 to 42.

46. A decoder, comprising:

a second determining unit configured to:

parse a bitstream to determine first syntax flag information;

construct a block vector list for a current block when the first syntax flag information indicates that a preset intra prediction mode is to be applied to the current block; and

determine a prediction block for the current block according to the block vector list for the current block; wherein the block vector list for the current block satisfies at least one of:

the block vector list is constructed by performing an auto-relocated block vector prediction (AR-BVP) process at least twice, and between the at least two AR-BVP processes, checking is performed on at least one non-AR-BVP block vector;

the block vector list comprises a block vector for a reference block corresponding to a guiding block vector for the current block;

the block vector list comprises a subsample-precision block vector; wherein the subsample-precision block vector is used for decoder-side intra mode derivation (DIMD), or template-based intra mode derivation (TIMD), or spatial geometric partitioning mode (SGPM); or

the block vector list is obtained by sorting template error values corresponding to block vectors in the block vector list; wherein the template error values corresponding to the block vectors are determined in a preset template-error calculation method, and the preset template-error calculation method is determined according to an error calculation method

for a template search process corresponding to the current block or second syntax flag information transmitted in the bitstream.

47. A decoder, comprising a second memory and a second processor, wherein:

the second memory is configured to store computer programs executable by the second processor; and

the second processor is configured to, when executing the computer programs, perform the method of any of claims 1 to 21.

48. A computer-readable storage medium, configured to store computer programs, wherein the computer programs, when executed by at least one processor, are operable to implement the method of any of claims 1 to 21 or the method of any of claims 22 to 42.