(12) STANDARD PATENT (11) Application No. **AU 2015204953 B2**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Representing motion vectors in an encoded bitstream

(51) International Patent Classification(s)
H04N 19/523 (2014.01) ***H04N 19/52*** (2014.01)
H04N 19/43 (2014.01) ***H04N 19/70*** (2014.01)
H04N 19/513 (2014.01)

(21) Application No: **2015204953** (22) Date of Filing: **2015.01.07**

(87) WIPO No: **WO15/105816**

(30) Priority Data

(31) Number	(32) Date	(33) Country
61/925,108	**2014.01.08**	**US**
61/934,506	**2014.01.31**	**US**
14/530,625	**2014.10.31**	**US**

(43) Publication Date: **2015.07.16**
(44) Accepted Journal Date: **2018.11.01**

(71) Applicant(s)
Microsoft Technology Licensing, LLC

(72) Inventor(s)
Silkin, Sergey;Sablin, Sergey;Zhou, You;Lin, Chih-Lung;Lee, Ming-Chieh;Sullivan, Gary J.

(74) Agent / Attorney
Davies Collison Cave Pty Ltd, Level 15 1 Nicholson Street, MELBOURNE, VIC, 3000, AU

(56) Related Art
US 7224731 B2
US 2011/0243234 A1

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 July 2015 (16.07.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/105816 A3

(51) International Patent Classification:
H04N 19/523 (2014.01) *H04N 19/513* (2014.01)
H04N 19/70 (2014.01) *H04N 19/43* (2014.01)
H04N 19/52 (2014.01)

(21) International Application Number:
PCT/US2015/010391

(22) International Filing Date:
7 January 2015 (07.01.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/925,108	8 January 2014 (08.01.2014)	US
61/934,506	31 January 2014 (31.01.2014)	US
14/530,625	31 October 2014 (31.10.2014)	US

(71) Applicant: **MICROSOFT TECHNOLOGY LICENSING, LLC** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).

(72) Inventors: **SILKIN, Sergey**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US). **SABLIN, Sergey**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US). **ZHOU, You**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US). **LIN, Chih-Lung**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US). **LEE, Ming-Chieh**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US). **SULLIVAN, Gary J.**; c/o Microsoft Corporation, LCA - International Patents (8/1172), One Microsoft Way, Redmond, Washington 98052-6399 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,

[Continued on next page]

(54) Title: REPRESENTING MOTION VECTORS IN AN ENCODED BITSTREAM

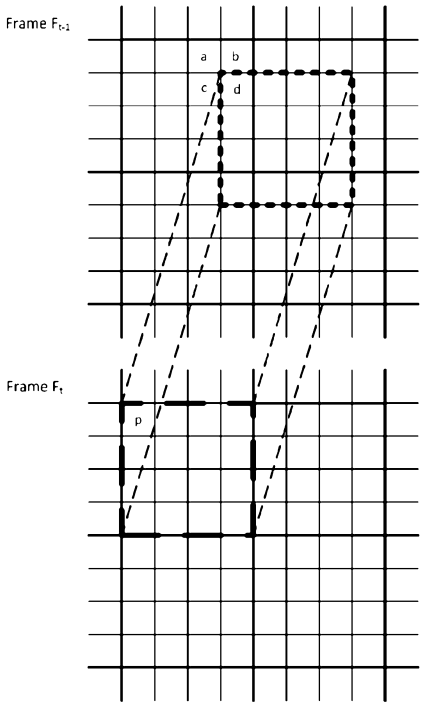

Figure 7

(57) Abstract: A format for use in encoding moving image data, comprising: a sequence of frames including plurality of the frames in which at least a region is encoded using motion estimation; a respective set of motion vector values representing motion vectors of the motion estimation for each respective one of these frames or each respective one of one or more regions within each of such frames; and at least one respective indicator associated with each of the respective frames or regions, indicating whether the respective motion vector values of the respective frame or region are encoded at a first resolution or a second resolution.

WO 2015/105816 A3

HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
28 April 2016

REPRESENTING MOTION VECTORS IN AN ENCODED BITSTREAM

BACKGROUND

[0001] In modern communication systems a video signal may be sent from one terminal to another over a medium such as a wired and/or wireless network, often a packet-based network such as the Internet. For example the video may be part of a VoIP (voice over Internet Protocol) call conducted from a VoIP client application executed on a user terminal such as a desktop or laptop computer, tablet or smart phone.

[0002] Typically the frames of the video are encoded by an encoder at the transmitting terminal in order to compress them for transmission over the network. The encoding for a given frame may comprise intra frame encoding whereby blocks are encoded relative to other blocks in the same frame. In this case a target block is encoded in terms of a difference (the residual) between that block and a neighbouring block. Alternatively the encoding for some frames may comprise inter frame encoding whereby blocks in the target frame are encoded relative to corresponding portions in a preceding frame, typically based on motion prediction. In this case a target block is encoded in terms of a motion vector identifying an offset between the block and the corresponding portion from which it is to be predicted, and a difference (the residual) between the block and the corresponding portion from which it is predicted. A corresponding decoder at the receiver decodes the frames of the received video signal based on the appropriate type of prediction, in order to decompress them for output to a screen at the decoder side.

[0003] When encoding (compressing) a video, the motion vectors are used to generate the inter frame prediction for the current frame. The encoder first searches for a similar block (the reference block) in a previous encoded frame that best matches the current block (target block), and signals the displacement between the reference block and target block to the decoder as part of the encoded bitstream. The displacement is typically represented as horizontal and vertical x and y coordinates, and is referred to as the motion vector.

[0004] The reference "block" is not in fact constrained to being at an actual block position in the reference frame, i.e. is not restricted to the same grid as the target blocks, but rather it is a correspondingly-sized portion of the reference frame offset relative to the target block's position by the motion vector. According to present standards the motion vectors are represented at fractional pixel resolution. For instance in the H.264 standard each motion vector is represented at ¼ pixel resolution. So by way of example, if a 16x16

2015204953 05 Sep 2018

block in the current frame is to be predicted from another 16x16 block in the previous frame that is at 1 pixel left of the position of the target block, then the motion vector is (4,0). Or if the target block is to be predicted from a reference block that is only, say, ¾ of a pixel to the left of the target block, the motion vector is (3,0). The reference block at a fractional pixel position does not actually exist per se, but rather it is generated by interpolation between pixels of the reference frame. The sub-pixel motion vectors can achieve significant performance in terms of compression efficiency.

[0004A] It is desired to provide a decoder, a computer program product embodied on a computer-readable storage hardware memory device, and a network hardware device that alleviate one or more difficulties of the prior art, or that at least provides a useful alternative.

SUMMARY

[0004B] In accordance with some embodiments of the present invention, there is provided a decoder comprising:

an input receiving moving image data in encoded form, including a plurality of frames in which a region is encoded using motion estimation, the moving image data including a respective set of motion vector values associated with each respective one of the frames or with each respective region within each of the frames, each of the motion vector values of each respective frame or region is included in a motion vector field in an encoded bitstream, the moving image data further including at least one respective indicator in header information associated with each of the respective frames or regions indicating whether the respective motion vector values are encoded at a first resolution or a second resolution, the motion vector field having a reduced size for the frames or the regions having the motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution, and the moving image data including a setting to set whether the first resolution or the second resolution of the motion vector values is being indicated per region or per frame; and

a motion prediction module decoding the frames or regions based on the motion vector values, the decoding comprising:

reading each of the indicators from the header information;

reading the setting and interpreting the motion vector values accordingly;

2015204953 05 Sep 2018

determining, from the read indicators, whether the motion vector values of the respective frame or region are encoded at the first resolution or the second resolution; and

interpreting the motion vector values in units of fractional pixels for the indicated first resolution or in units of integer pixels responsive for the indicated second resolution.

[0004C] In accordance with some embodiments of the present invention, there is provided a computer program product embodied on a computer-readable storage hardware memory device that, responsive to execution by a processor performs operations comprising:

receiving moving image data in encoded form, including a plurality of frames in which a region is encoded using motion estimation, the moving image data including a respective set of motion vector values associated with each respective one of the frames or with each respective region within each of the frames, each of the motion vector values of each respective frame or region is included in a motion vector field in an encoded bitstream, the moving image data further including at least one respective indicator in header information associated with each of the respective frames or regions indicating whether the respective motion vector values are encoded at a first resolution or a second resolution, the motion vector field having a reduced size for the frames or the regions having the motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution, and the moving image data including a setting to set whether the first resolution or the second resolution of the motion vector values is being indicated per region or per frame;

reading each of the indicators from the header information;

reading the setting and interpreting the motion vector values accordingly;

determining, from the read indicators, whether the motion vector values of the respective frame or region are encoded at the first resolution or the second resolution;

interpreting the respective motion vector values in units of fractional pixels for the indicated first resolution or in units of integer pixels for the indicated second resolution; and

decoding the frames or regions based on the motion vector values as interpreted in the units of fractional or integer pixels in accordance with the interpreting.

[0004D] In accordance with some embodiments of the present invention, there is provided a network hardware device comprising: a computer-readable storage hardware memory

2015204953 05 Sep 2018

device having stored thereon a program, that responsive to execution by a processor, generates an encoded bitstream, the encoded bitstream comprising:

a sequence of frames including a plurality of frames in which a region is encoded using motion estimation;

a respective set of motion vector values representing motion vectors of the motion estimation for each respective one of said frames or each respective region within each of said frames, each of the motion vector values of each respective frame or region is included in a motion vector field in the encoded bitstream, the respective set of motion vector values encoded according to a protocol whereby motion vector values encoded at a first resolution represent fractional pixel units and motion vector values encoded at a second resolution represent integer pixel units, wherein according to said protocol, the motion vector field has a reduced size for frames or regions having motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution; and

a respective indicator in header information associated with each of said respective frames or regions, indicating whether the respective motion vector values of the respective frame or region are encoded at the first resolution or the second resolution as indicated per region or per frame, and enabling a decoder to read the indicator and to interpret the respective motion vector values in units of fractional pixels or integer pixels units by including the indicator in the header information.

[0005] Using a fractional pixel resolution incurs more bits to encode the motion vector than if motion was estimated at integer pixel resolution, and it also incurs more processing resources in searching for the best matching reference. For video coding this may be worthwhile, e.g. as the reduced size of a better-matched residual may generally outweigh the bits incurred encoding the motion vector, or the quality achieved may be considered to justify the resources. However, not all moving images to be encoded are videos (i.e. captured from a camera). It is recognised herein that when encoding (compressing) a moving image that is captured from a screen rather than a camera, most of the motion vectors in the encoded bit stream will generally point to integer pixels, while very few of them tend to be found at fractional pixel positions. Thus while encoders normally represent motion vectors in bit streams in units of ¼ pixels, for screen sharing or recording applications bandwidth can in fact be saved without undue loss of quality by encoding the motion vectors in units of only 1 pixel.

2015204953 05 Sep 2018

[0006] Nonetheless, considering that the fractional motion vector can still be useful for normal video (captured by camera) or perhaps other moving images (e.g. animations), the motion vector may be signalled in a flexible way: when the video source is from a captured screen the motion vector may be signalled in units of 1 pixel, but for normal video and/or other moving images a fractional pixel unit may still be used.

[0007] More generally, there may be various circumstances in which it may be useful to have control over whether fractional or integer pixel motion vector resolution is used, e.g. depending on how the designer of the encoder wishes to implement any desired trade off or effect. E.g. perhaps some video or animations due to some aspect of their nature will be more efficiently served by integer pixel resolution in the motion estimation, while other videos or other types of moving image may be more efficiently served by fractional pixel resolution.

[0008] Also disclosed herein is a format for use in encoding moving image data, whereby moving image data encoded according to said format comprises:

- a sequence of frames including plurality of said frames in which at least a region is encoded using motion estimation;
- a respective set of motion vector values representing motion vectors of the motion estimation for each respective one of said frames, or each respective one of one or more regions within each of said frames; and
- at least one respective indicator associated with each of said respective frames or regions, indicating whether the respective motion vector values of the respective frame or region are encoded at a first resolution or a second resolution.

[0009] The motion vector values are encoded according to a protocol whereby motion vector values encoded at the first resolution are represented on a scale having a larger number of finer steps, and motion vector values encoded at the second resolution are represented on a scale having a smaller number of coarser steps and thereby incur fewer bits on average in the encoded bitstream. The coarser steps represent integer pixel units and the finer steps represent fractional pixel units.

[0010] Also disclosed herein is a network element or computer-readable storage medium carrying bitstream of moving image data encoded according to such a format or protocol.

2015204953 05 Sep 2018

[0011] There may be provided a bitstream comprising some of said plurality of frames or regions encoded at the first resolution and others of said plurality of frames or regions encoded at the second resolution, the respective indicator indicating the resolution individually for each of said plurality of (inter frame encoded) frames or regions.

[0012] Each of the motion vector values of each frame or region may be included in a motion vector field of the encoded bitstream, and according to said protocol the motion vector field may have a reduced size for frames or regions whose motion vectors are encoded at the second resolution.

[0013] Also disclosed herein is a decoder comprising an input for receiving moving image data in encoded form, and a motion prediction module. The moving image data includes a plurality of frames in which at least a region is encoded using motion estimation (i.e. inter frame encoded frames), based on a format or protocol in accordance with any of the embodiments disclosed herein. The motion prediction module decodes said (inter frame encoded) frames or regions based on the motion vector values. This includes reading each of the indicators to determine whether the motion vector values of the respective frame or region are encoded at the first or second resolution, and if the first resolution to interpret the motion vector values in units of fractional pixels, and if the second resolution to interpret the motion vector values in units of integer pixels.

[0014] The moving image data may comprise a respective two indicators associated with each of said frames or regions, the two indicators indicating the resolution of respective motion vectors in two dimensions, and the motion prediction module may be configured to read both indicators and interpret the respective motion vector values accordingly.

[0015] Each of at least some of said frames may be divided into multiple regions; the moving image data may comprise at least one respective indicator associated with each respective one of the multiple regions to individually indicate whether the motion vector values of the respective region are encoded at the first or second resolution; and the motion prediction module may be configured to read the indicators to determine whether the motion vector values of each respective region are encoded at the first or second resolution, and to interpret the respective motion vector values in said units of fractional pixels or integer pixels accordingly. In embodiment said regions may be slices of an H.26x standard.

[0016] The moving image data may further comprises a setting to set whether the resolution of the motion vector values is being indicated per region or per frame, and the

motion prediction module may be configured to read the setting and interpret the motion vector values accordingly.

[0017] The motion prediction module may be configured to interpret the respective motion vector values in units of fractional pixels as a default if the respective indicator is not present for one of said frames or regions.

[0018] The moving image data including the motion vectors may be further encoded according to a lossless encoding technique. The decoder may comprise an inverse of a lossless encoding stage preceding said decoding by the motion prediction module.

[0019] Also disclosed herein is a computer program product embodied on a computer-readable storage medium and configured so as when executed to perform operations of the decoder according to any of the embodiments disclosed herein.

[0020] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter. Nor is the claimed subject matter limited to implementations that solve any or all of the disadvantages noted herein.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] Some embodiments of the present invention are hereinafter disclosed, by way of example only, with reference to the accompanying drawings in which:

[0022] Figure 1 is a schematic representation of a video stream,

[0023] Figure 2 is a schematic block diagram of a communication system,

[0024] Figure 3 is a schematic representation of an encoded video stream,

[0025] Figure 4 is a schematic block diagram of an encoder,

[0026] Figure 5 is a schematic block diagram of a decoder,

[0027] Figure 6 is a schematic representation of inter frame encoding scheme, and

[0028] Figure 7 is a schematic representation of another inter frame encoding scheme.

DETAILED DESCRIPTION

[0029] Figure 1 gives a schematic illustration of an input video signal captured from a camera, and divided into spatial divisions to be encoded by a video encoder so as to generate an encoded bitstream. The signal comprises a moving video image divided in time into a plurality of frames (F), each frame representing the image at a different respective moment in time (…t-1, t, t+1…). Within each frame, the frame is divided in space into a plurality of divisions each representing a plurality of pixels. These divisions

2015204953 05 Sep 2018

may be referred to as blocks. In certain schemes, the frame is divided and sub-divided into different levels of block. For example each frame may be divided into macroblocks (MB) and each macroblock may be divided into blocks (b), e.g. each block representing a region of 8x8 pixels within a frame and each macroblock representing a region of 2x2 blocks (16x16 pixels). In certain schemes each frame can also be divided into independently decodable slices (S), each comprising a plurality of macroblocks. The slices S can generally take any shape, e.g. each slice being one or more rows of macroblocks or an

irregular or arbitrarily defined selection of macroblocks (e.g. corresponding to a region of interest, ROI, in the image).

[0030] With regard to the term "pixel", in the following the term is used to refer to samples and sampling positions in the sampling grid for the picture array (sometimes in the literature the term "pixel" is instead used to refer to all three colour components corresponding to one single spatial position, and sometimes it is used to refer to a single position or a single integer sample value in a single array). The resolution of the sampling grid is often different between the luma and chroma sampling arrays. In embodiments the following may be applied to a 4:4:4 representation, but it may potentially also be applied in 4:2:2 and 4:2:0 for example.

[0031] Note also that while any given standard may give specific meanings to the terms block or macroblock, the term block is also often used more generally in the art to refer to a division of the frame at a level on which encoding and decoding operations like intra or inter prediction are performed, and it is this more general meaning that will be used herein unless specifically stated otherwise. For example the blocks referred to herein may in fact be the divisions called blocks or macroblocks in the H.26x standards, and the various encoding and decoding stages may operate at a level of any such divisions as appropriate to the encoding mode, application and/or standard in question.

[0032] A block in the input signal as captured is usually represented in the spatial domain, where each colour-space channel is represented as a function of spatial position within the block. For example in YUV colour space each of the luminance (Y) and chrominance (U,V) channels may be represented as a function of Cartesian coordinates x and y, Y(x,y), U(x,y) and V(x,y); or in RGB colour space each of the red (R), green (G) and blue (B) channels may be represented as a function of Cartesian coordinates R(x,y), G(x,y), B(x,y). In this representation, each block or portion is represented by a set of pixel values at different spatial coordinates, e.g. x and y coordinates, so that each channel of the colour space is represented in terms of a respective magnitude of that channel at each of a discrete set of pixel locations.

[0033] Prior to quantization however, the block may be transformed into a transform domain representation as part of the encoding process, typically a spatial frequency domain representation (sometimes just referred to as the frequency domain). In the frequency domain each colour-space channel in the block is represented as a function of spatial frequency (dimensions of 1/length) in each of two dimensions. For example this could be denoted by wavenumbers k_x and k_y in the horizontal and vertical directions

respectively, so that the channels may be expressed as $Y(k_x, k_y)$, $U(k_x, k_y)$ and $V(k_x, k_y)$ in YUV space; or $R(k_x, k_y)$, $G(k_x,k_y)$, $B(k_x,k_y)$ in RGB space. Thus instead of representing a colour-space channel in terms of a magnitude at each of a discrete set of pixel positions, the transform represents each colour-space channel in terms of a coefficient associated with each of a discrete set of spatial frequency components which make up the block, i.e. an amplitude of each of a discrete set of spatial frequency terms corresponding to different frequencies of spatial variation across the block. Possibilities for such transforms include a Fourier transform, Discrete Cosine Transform (DCT), Karhunen-Loeve Transform (KLT), or others.

[0034] The block diagram of Figure 2 gives an example of a communication system in which the techniques of this disclosure may be employed. The communication system comprises a first, transmitting terminal 12 and a second, receiving terminal 22. For example, each terminal 12, 22 may comprise one of a mobile phone or smart phone, tablet, laptop computer, desktop computer, or other household appliance such as a television set, set-top box, stereo system, etc. The first and second terminals 12, 22 are each operatively coupled to a communication network 32 and the first, transmitting terminal 12 is thereby arranged to transmit signals which will be received by the second, receiving terminal 22. Of course the transmitting terminal 12 may also be capable of receiving signals from the receiving terminal 22 and vice versa, but for the purpose of discussion the transmission is described herein from the perspective of the first terminal 12 and the reception is described from the perspective of the second terminal 22. The communication network 32 may comprise for example a packet-based network such as a wide area internet and/or local area network, and/or a mobile cellular network.

[0035] The first terminal 12 comprises a computer-readable storage medium 14 such as a flash memory or other electronic memory, a magnetic storage device, and/or an optical storage device. The first terminal 12 also comprises a processing apparatus 16 in the form of a processor or CPU having one or more execution units, a transceiver such as a wired or wireless modem having a transmitter 18, a video camera 15 and a screen 17 (i.e. a display or monitor). Each of the camera 15 and screen 17 may or may not be housed within the same casing as the rest of the terminal 12 (and even the transmitter 18 could be internal or external, e.g. comprising a dongle or wireless router in the latter case). The storage medium 14, video camera 15, screen 17 and transmitter 18 are each operatively coupled to the processing apparatus 16, and the transmitter 18 is operatively coupled to the network 32 via a wired or wireless link. Similarly, the second terminal 22 comprises a computer-

readable storage medium 24 such as an electronic, magnetic, and/or an optical storage device; and a processing apparatus 26 in the form of a CPU having one or more execution units. The second terminal comprises a transceiver such as a wired or wireless modem having at least a receiver 28 and a screen 25 which may or may not be housed within the same casing as the rest of the terminal 22. The storage medium 24, screen 25 and receiver 28 of the second terminal are each operatively coupled to the respective processing apparatus 26, and the receiver 28 is operatively coupled to the network 32 via a wired or wireless link.

[0036] The storage 14 on the first terminal 12 stores at least an encoder for encoding moving image data, the encoder being arranged to be executed on the respective processing apparatus 16. When executed the encoder receives a "raw" (unencoded) input video stream from the video camera 15, it is operable to encode the video stream so as to compress it into a lower bitrate stream, and outputs the encoded video stream for transmission via the transmitter 18 and communication network 32 to the receiver 28 of the second terminal 22. The storage 24 on the second terminal 22 stores at least a video decoder arranged to be executed on its own processing apparatus 26. When executed the decoder receives the encoded video stream from the receiver 28 and decodes it for output to the screen 25.

[0037] The encoder and decoder are also operable to encode and decode other types of moving image data, including screen sharing streams. A screen sharing stream is image data captured from a screen 17 at the encoder side so that one or more other, remote users can see what the user at the encoder side is seeing on screen, or so the user of that screen can record what's happening on screen for playback to one or more other users later. In the case of a call conducted between a transmitting terminal 12 and receiving terminal 22, the moving content of the screen 17 at the transmitting terminal 12 will be encoded and transmitted live (in real-time) to be decoded and displayed on the screen 25 of the receiving terminal 22. For example the encoder-side user may wish to share with another user how he or she is working the desktop of his or her operating system, or some application.

[0038] Note that where it is said that a screen sharing stream is captured from a screen, or the like, this does not limit to any particular mechanism for doing so. E.g. the data could be read from a screen buffer of the screen 17, or captured by receiving an instance of the same graphical data that is being output from the operating system or from an application for display on the screen 17.

[0039] Figure 3 gives a schematic representation of an encoded bitstream 33 as would be transmitted from the encoder running on the transmitting terminal 12 to the decoder running on the receiving terminal 22. The bitstream 33 comprises encoded image data 34 for each frame or slice comprising the encoded samples for the blocks of that frame or slice along with any associated motion vectors In one application, the bitstream may be transmitted as part of a live (real-time) call such as a VoIP call between the transmitting and receiving terminals 12, 22 (VoIP calls can also include video and screen sharing). The bitstream 33 also comprises header information 36 associated with each fame or slice. In embodiments the header 36 is arranged to include at least one additional element in the form of at least one flag 37 indicating the resolution of the motion vector, which will be discussed in more detail below.

[0040] Figure 4 is a block diagram illustrating an encoder such as might be implemented on transmitting terminal 12. The encoder comprises a main encoding module 40 comprising: a discrete cosine transform (DCT) module 51, a quantizer 53, an inverse transform module 61, an inverse quantizer 63, an intra prediction module 41, an inter prediction module 43, a switch 47, a subtraction stage (-) 49, and a lossless decoding stage 65. The encoder further comprises a control module 50 coupled to the inter prediction module 43. Each of these modules or stages may be implemented as a portion of code stored on the transmitting terminal's storage medium 14 and arranged for execution on its processing apparatus 16, though the possibility of some or all of these being wholly or partially implemented in dedicated hardware circuitry is not excluded.

[0041] The subtraction stage 49 is arranged to receive an instance of the input signal comprising a plurality of blocks over a plurality of frames (F). The input stream is received from a camera 15 or captured from what is being displayed on the screen 17. The intra or inter prediction 41, 43 generates a predicted version of a current (target) block to be encoded based on a prediction from another, already-encoded block or correspondingly-sized reference portion. The predicted version is supplied to an input of the subtraction stage 49, where it is subtracted from the input signal (i.e. the actual signal) in the spatial domain to produce a residual signal representing a difference between the predicted version of the block and the corresponding block in the actual input signal.

[0042] In intra prediction mode, the intra prediction 41 module generates a predicted version of the current (target) block to be encoded based on a prediction from another, already-encoded block in the same frame, typically a neighbouring block. When performing intra frame encoding, the idea is to only encode and transmit a measure of how

a portion of image data within a frame differs from another portion within that same frame. That portion can then be predicted at the decoder (given some absolute data to begin with), and so it is only necessary to transmit the difference between the prediction and the actual data rather than the actual data itself. The difference signal is typically smaller in magnitude, so takes fewer bits to encode (due to the operation of the lossless compression stage 65 – see below).

[0043] In inter prediction mode, the inter prediction module 43 generates a predicted version of the current (target) block to be encoded based on a prediction from another, already-encoded reference portion in a different frame than the current block, the reference portion having the size of a block but being offset relative to the target block in the spatial domain by a motion vector that is predicted by the inter prediction module 43 (inter prediction may also be referred to as motion prediction or motion estimation). The inter-prediction module 43 selects the optimal reference for a given target block by searching, in the spatial domain, through a plurality of candidate reference portions offset by a plurality of respective possible motion vectors in one or more frames other than the target frame, and selecting the candidate that minimises the residual with respect to the target block according to a suitable metric. The inter prediction module 43 is switched into the feedback path by switch 47, in place of the intra frame prediction stage 41, and so a feedback loop is thus created between blocks of one frame and another in order to encode the inter frame relative to those of the other frame. I.e. the residual now represents the difference between the inter predicted block and the actual input block. This typically takes even fewer bits to encode than intra frame encoding.

[0044] The samples of the residual signal (comprising the residual blocks after the predictions are subtracted from the input signal) are output from the subtraction stage 49 through the transform (DCT) module 51 (or other suitable transformation) where their residual values are converted into the frequency domain, then to the quantizer 53 where the transformed values are converted to substantially discrete quantization indices. The quantized, transformed indices of the residual as generated by the transform and quantization modules 51, 53, as well as an indication of the prediction used in the prediction modules 41,43 and any motion vectors generated by the inter prediction module 43, are all output for inclusion in the encoded video stream 33 (see element 34 in Figure 3); via a further, lossless encoding stage 65 such as a Golomb encoder or entropy encoder where the motion vectors and transformed, quantized indices are further compressed using lossless encoding techniques known in the art.

[0045] An instance of the quantized, transformed signal is also fed back though the inverse quantizer 63 and inverse transform module 61 to generate a predicted version of the block (as would be seen at the decoder) for use by the selected prediction module 41 or 43 in predicting a subsequent block to be encoded, in the same way the current target block being encoded was predicted based on an inverse quantized and inverse transformed version of a previously encoded block. The switch 47 is arranged to pass the output of the inverse quantizer 63 to the input of either the intra prediction module 41 or inter prediction module 43 as appropriate to the encoding used for the frame or block currently being encoded.

[0046] Figure 5 is a block diagram illustrating a decoder such as might be implemented on the receiving terminal 22. The decoder comprises an inverse of the lossless encoding 95, an inverse quantization stage 83, an inverse DCT transform stage 81, a switch 70, and an intra prediction stage 71 and a motion compensation stage 73. Each of these modules or stages may be implemented as a portion of code stored on the receiving terminal's storage medium 24 and arranged for execution on its processing apparatus 26, though the possibility of some or all of these being wholly or partially implemented in dedicated hardware circuitry is not excluded.

[0047] The inverse quantizer 81 is arranged to receive the encoded signal 33 from the encoder, via the receiver 28 and inverse lossless coding stage 95. The inverse quantizer 81 converts the quantization indices in the encoded signal into de-quantized samples of the residual signal (comprising the residual blocks) and passes the de-quantized samples to the reverse DCT module 81 where they are transformed back from the frequency domain to the spatial domain. The switch 70 then passes the de-quantized, spatial domain residual samples to the intra or inter prediction module 71 or 73 as appropriate to the prediction mode used for the current frame or block being decoded, and the intra or inter prediction module 71, 73 uses intra or inter prediction respectively to decode the blocks. Which mode to use is determined using the indication of the prediction and/or any motion vectors received with the encoded samples 34 in the encoded bitstream 33. Following on from this stage, the decoded blocks are output to be played out through the screen 25 at the receiving terminal 22.

[0048] As mentioned, codecs according to conventional standards perform motion prediction at a resolution of quarter pixels, meaning the motion vectors are also expressed in terms of quarter pixel steps. An example of quarter pixel resolution motion estimation is shown in Figure 6. In this example, pixel p in the upper left corner of the target block is

predicted from an interpolation between the pixels a, b, c and d, and the other pixels of the target block will also be predicted based on a similar interpolation between respective groups of pixels in the reference frame, according to the offset between the target block in one frame and the reference portion in the other frame (these blocks being shown with bold dotted lines in Figure 6). However, performing motion estimation with this granularity has consequences, as discussed below.

[0049] Referring to the lossless coder 65 and decoder 95, lossless coding is a form of compression which works not by throwing away information (like quantization), but by using different lengths of codeword to represent different values depending on how likely those values are to occur, or how frequently they occur, in the data to be encoded by the lossless encoding stage 65. For example the number of leading 0s in the codeword before encountering a 1 may indicate the length of the codeword, so 1 is the shortest codeword, then 010 and 011 are the next shortest, then 00100 …, and so forth. Thus the shortest codewords are much shorter than would be required if a uniform codeword length was used, but the longest are longer than that. But by allocating the most frequent or likely values to the shortest codewords and only the least likely or frequently occurring values to the longer codewords, the resulting bitstream 33 can on average incur fewer bits per encoded value than if a uniform codeword length was used, and thus achieve compression without discarding any further information.

[0050] Much of the encoder 40 prior to the lossless encoding stage 65 is designed to try to make as many of the values as small as possible before being passed through the lossless coding stage 65. As they then occur more often, smaller values will then incur lower bitrate in the encoded bitstream 33 than larger values. This is why the residual is encoded as opposed to absolute samples. It is also the rationale behind the transform 51, as many samples tend to transform to zero or small coefficients in the transform domain.

[0051] A similar consideration can be applied to the encoding of the motion vectors.

[0052] For instance, in H.264/MPEG-4 Part 10 and H.265/HEVC the motion vector is encoded with Exponential Golomb Coding. The following table shows the motion vector values and the encoded bits.

Value	Codeword	Number of Bits Incurred
0	1	1
1	010	3
2	011	3
3	00100	5
4	00111	5
5	0001000	7
...		

[0053] From the table above it can be seen that the larger the value is, the more bits are used. This means the higher the resolution of the motion vector, the more bits are incurred. E.g. so with a quarter pixel resolution, an offset of 1 pixel has to be represented by a value of 4, incurring 5 bits in the encoded bitstream.

[0054] In encoding video (captured from a camera) the cost of this resolution in the motion vector may be worthwhile, as the finer resolution may provide more opportunities in the search for a lower cost residual reference. However, it is observed herein that for moving images captured from a screen, most of the spatial displacements tend to be at full pixel displacements and few of them tend to be at fractional pixel positions, so most of the motion vectors tend to point to integer pixel values and very few tend to point to fractional pixel values.

[0055] On such a basis, it may be desirable to encode the motion vectors for image data captured from a screen with a resolution of 1 pixel. Considering the fact that no bits need to be spent on the fractional parts of motion vectors for such content, this means the bit rate incurred in encoding such content can be reduced.

[0056] For example, while encoders normally interpret motion vectors in bitstreams in units of ¼ pixel offsets, an encoder may in fact often be able to save bit rate by abandoning this resolution and instead encoding the motion vectors for screen coding applications in units of integer pixel offsets. Although it will reduce the precision of the motion vectors by a factor of four, such precision is generally less worthwhile for screen sharing or recording applications and this also reduces the number of bits needed to code the vectors. To predict a current (target) block from a reference block 1 pixel left of the target block, the motion vector will be (1,0) instead of (4,0). Using the above Golomb encoding, this means the bits incurred for encoding the motion vector change from (00111, 1) to (010, 1) and so two bits are saved in this case.

[0057] Furthermore, in embodiments the reduced resolution motion vector may also reduce the complexity of the motion estimation performed at the encoder by restricting the motion vector search to integer values, thus reducing processing resources incurred by the search. Alternatively it would be possible to perform a normal search and round the resulting motion vectors to integer values.

[0058] Figure 7 shows an example of motion prediction constrained to a resolution of whole pixels only, with the motion vector being constrained to whole pixel steps only. In contrast with Figure 6, the pixel p is predicted only from a single, whole pixel a with no interpolation. It could alternatively have been predicted from pixel b, c, d or another pixel depending on the offset between the target block in one frame and the reference portion in the other frame (shown again with bold dotted lines), but due to the constraint it could not have been predicted from an interpolation between pixels. Note: for any given block the quarter pixel prediction as illustrated by way of example in Figure 6 could happen to have generated a whole pixel offset with no interpolation, if that gave the lowest residual. However, it would not have been constrained to doing so, and over a sizable image it would be highly unlikely that would have happened for all blocks.

[0059] Considering that fractional motion vector values can still be very useful for camera-captured content, in embodiments the encoder 40 is provided with a controller 50 coupled to the motion prediction module 43 with the controller 50 being configured to select the motion vector resolution in a flexible way: when the source data is from a captured screen 17 and there is no fractional pixel motion, the motion vector is encoded and transmitted in units of only whole pixels; but for camera-content video the motion vectors are still encoded and transmitted with fractional pixel precision.

[0060] In order to do this, the controller 50 may be configured to measure a performance heuristic indicative of the fact that the type of content being captured is screen content. In response, it then disables fractional pixel motion compensation for screen content coding. Alternatively, the controller 50 may receive an indication from an application or the operating system as to what type of data it is supplying to the encoder for encoding, and the controller 50 may select between the mode on that basis. As another option is may make the selection based on historical data. The selection may be made on a per-frame basis, or the mode may be selected individually for different regions within a frame, e.g. on a per slice basis.

[0061] Thus before encoding a frame or slice, the encoder is able to decide the motion vector resolution based on factors such as historical statistical data, knowledge of its type

of application, multi-pass analysis, or some other such technique. If the encoder decides to use full pixel motion estimation only, the fractional pixel search is skipped. If a scaled motion vector prediction has a fractional part, the prediction may be rounded to an integer value.

[0062] In further embodiments, the control may optionally be applied separately to the vertical or horizontal component of a vector. This may be useful for encoding screen video that is scaled horizontally or vertically. E.g. consider a case where the encoder is working with a screen share resized horizontally or vertically. In this case, one of the components of the motion vectors will have fractional parts while the other component will not. To control the bit rate, the encoder may adjust the quantization parameter (QP) in a predefined range. If the QP has reached to the allowed upper bound, and the bit rate is still too high, the encoder may trigger an internal resizer. It resizes the incoming pictures horizontally and at the decoder side the decoded pictures are stretched back to original width. Hence a situation may arise where it would also be useful to treat the horizontal and vertical motion vectors differently.

[0063] In order to represent the motion vector on a reduced resolution scale in units or steps of integer pixels, and thus achieve the associated bitrate savings over conventional codecs, the protocol for signalling the motion vectors will have to be updated for future codec standards. In embodiments this may be implemented as an update to the H.265 (HEVC, High Efficiency Video Coding) standard. For encoding captured screen content, the format of the encoded data 34 will be given a reduced size motion vector field for each motion vector. For an encoded screen capture stream encoded in the integer pixel mode, the relevant data 34 will thus comprise integer motion vectors in the bitstream 33 and in embodiments only integer motion vectors in the bitstream 33.

[0064] In embodiments this will be optional, with a flag 37 also included in the header 36 to indicate whether fractional pixel (e.g. ¼ pixel) or integer pixel resolution is being used in the encoding of the associated frame or slice (refer again to Figure 3). If the horizontal and vertical resolutions can be selected separately, two flags 37 will be required per frame or slice.

[0065] The following discloses an exemplary embodiment based on an update to the H.265 standard. The modification enables motion vectors to be represented on a reduced, integer pixel scale in the encoded bitstream 33, and adds two flags 37 per slice in the header information 36 of the compressed stream in order to signal the resolution of motion vectors in their horizontal and vertical components.

[0066] The modification does not change the syntax or parsing process other than at the header level, but modifies the decoding process by interpreting the motion vector differences as integers and rounding scaled MV predictors to integer values. The modification has been found to increase coding efficiency as much as 7% and on average by about 2% for tested screen content sequences, and it can also reduce the complexity of the encoding and decoding processes.

[0067] A high-level indicator is added (at the SPS, PPS, and/or slice header level) to indicate the resolution for interpretation of the motion vectors.

[0068] In the decoding process, if the motion vectors are indicated to be at full pixel resolution and a scaled motion vector prediction has a fractional part, then in embodiments the prediction is rounded to an integer value. Motion vector differences are simply interpreted as integer offsets rather than ¼-sample offsets. All other decoding processes remain the same. The parsing process (below the header level) is also unchanged. When the motion vectors are coded at full-sample precision and the input image data uses 4:2:2 or 4:2:0 sampling, the chroma motion vectors can be derived in the usual manner, which will produce ½-sample chroma motion displacements. Alternatively, the chroma motion vectors may also be rounded to integer values.

[0069] Regarding the syntax change for signalling the motion vector resolution, in embodiments there are three possible modes, with a mode indicator being signalled at a higher (per frame) header level such as the picture parameter set (PPS), or alternatively at an even higher level such as the sequence parameter set (SPS). The mode indicator has three selectable values. One value indicates that motion vectors are always interpreted as representing ¼ sample offsets (in pictures that use this PPS). A second value indicates that motion vectors are always interpreted as representing integer offsets (in pictures that use this PPS). A third value indicates that there is a signal at a lower level that adaptively selects the interpretation to be applied at that lower level. This lower-level indication may be a flag in the slice header that will be present only when operating in this third mode.

[0070] To implement this, in embodiments a new two-bit indicator will be included in the PPS extension to indicate the motion vector resolution control modes. This indicator may be referred to as motion_vector_resolution_control_idc. When the mode is 0, the motion vectors are encoded at ¼ pixel precision and all decoding processes remain unchanged. When the mode is 1, all of the motion vectors in the slices that refer to the PPS are encoded at full pixel precision. And when the mode is 2, the motion vector resolution

is controlled on a slice-by-slice basis by a flag in the slice header. When motion_vector_resolution_control_idc is not present, its value is inferred as 0.

When motion_vector_resolution_control_idc is equal to 2, an additional flag called slice_motion_vector_resolution_flag is signalled in the slice header. When the flag is zero, the motion vectors of this slice are encoded at ¼ pixel precision, and when the flag is 1, the motion vectors are encoded at full pixel precision. When the flag is not present, its value is inferred as equal to the value of motion_vector_resolution_control_idc.

The modified PPS syntax is illustrated as follows:

pic_parameter_set_rbsp( ) {	**Descriptor**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
dependent_slice_segments_enabled_flag	u(1)
output_flag_present_flag	u(1)
num_extra_slice_header_bits	u(3)
sign_data_hiding_enabled_flag	u(1)
cabac_init_present_flag	u(1)
…	
lists_modification_present_flag	u(1)
log2_parallel_merge_level_minus2	ue(v)
slice_segment_header_extension_present_flag	u(1)
pps_extension1_flag	u(1)
if(pps_extension1_flag) {	
if(transform_skip_enabled_flag)	
log2_max_transform_skip_block_size_minus2	ue(v)
luma_chroma_prediction_enabled_flag	u(1)
motion_vector_resolution_control_idc	u(2)
chroma_qp_adjustment_enabled_flag	u(1)
if(chroma_qp_adjustment_enabled_flag) {	
diff_cu_chroma_qp_adjustment_depth	ue(v)
chroma_qp_adjustment_table_size_minus1	ue(v)
for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
cb_qp_adjustment[i]	se(v)
cr_qp_adjustment[i]	se(v)
}	
}	
pps_extension2_flag	u(1)
}	
if(pps_extension2_flag)	

while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

The modified slice header syntax is illustrated as follows:

slice_segment_header( ) {	**Descriptor**
first_slice_segment_in_pic_flag	u(1)
if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
no_output_of_prior_pics_flag	u(1)
slice_pic_parameter_set_id	ue(v)
…	
if(slice_type = = P || slice_type = = B) {	
if(motion_vector_resolution_control_idc = = 2)	
slice_motion_vector_resolution_flag	u(1)
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag) {	
…	

[0071] Regarding the scaling mentioned above, this is something that can occur for example in HEVC (H. 265). The idea is that if a motion vector is used for coding some other frame, it can be computed what would be the motion vector that would be equivalent in terms of the relative positioning displacement between: (i) the current picture and (ii) its reference picture. This is based on the relative positioning of the displacement indicated by a motion vector in the co-located part of another picture, and based on the relative positioning displacement between (iii) that picture and (iv) the picture it was referencing as its reference picture. That is, the motion vectors themselves can be encoded based on prediction, with the motion vector for the target block in the current, target frame being encoded relative to a previous motion vector for a spatially corresponding block in a previous frame. The motion vector for the target block is then signalled in the encoded bitstream 33 only in terms of a difference (e.g. typically a delta) between the predicted

version and the actual version. The decoder can then recreate the encoded motion vector by performing the same prediction at the decoder side and adding the delta.

[0072] However, the temporal frame rate of the coded data is not always constant, and also there may be a difference between the order in which pictures are coded in the bitstream and the order in which they are captured and displayed. These temporal relationships may be computed and then used to scale the motion vector so that it basically represents the same speed of motion in the same direction. I.e. the predicted version of the motion vector is not just equal to the reference vector, but rather is a scaled version of it. This is known as temporal motion vector prediction.

[0073] The current motion vector (MV) decoding process in the HEVC standard may be summarized as follows.

1. Determine how the MV is to be predicted at the decoder and whether or not a motion vector delta (MVD) has been sent in the encoded bitstream from the encoder (this may involve some syntax indications).
2. Generate the predicted motion vector (PMV). This creates a pair of integers (px, py). The integers are assumed to represent offsets with quarter sample position precision in the luma domain.
3. If no MVD is sent, set the MV value (mx, my) to (px, py); otherwise decode the pair of integers (px, py) as (dx, dy), and set the MV (mx, my) to (px + dx, py + dy). The MV is interpreted as representing 1/4-sample offsets in the luma domain. Note: the luma distinction only matters if the video is in a format such as 4:2:2 or 4:2:0 which use different resolution in the luma channel than in the chroma channels; if instead the video is 4:4:4 format for example, the offset is interpreted as having 1/4 sample units for all sampling grids.
4. Store the final value (mx, my) for subsequent use for generating subsequent PMVs (and controlling deblocking filtering, etc.).

[0074] In embodiments of the present disclosure, when the integer pixel motion vector mode is being used, the scaled motion vector may be rounded in the decoding process to accommodate the restriction to integer pixel resolution. For example motion vector decoding process may be modified as follows.

1. Determine how the MV is to be predicted at the decoder and whether or not a motion vector delta (MVD) has been sent in the encoded bitstream from the encoder (this may involve some syntax indications).

2. Generate the predicted motion vector (PMV). This creates a pair of integers (px, py). The integers are assumed to represent offsets with quarter sample position precision in the luma domain.
3. If operating in integer-precision mode, do this:
    a. Round off the prediction value to make it correspond to an integer offset – e.g. create (rx, ry) = (round(px/4), round(py/4)).
    b. If no MVD is sent, set the MV value (mx, my) to (4*rx, 4*ry); otherwise decode the pair of integers (px, py) as (dx, dy), and set the MV (mx, my) to (4*(rx + dx), 4*(ry + dy)). The MV is then interpreted as representing 1/4-sample offsets in the luma domain (N.B. again the luma distinction only matters if the video is in a format such as 4:2:2 or 4:2:0).
4. Otherwise (i.e. when operating in 1/4-sample precision mode), do this: If no MVD is sent, set the MV value (mx, my) to (px, py); otherwise decode the pair of integers (px, py) as (dx, dy), and the MV (mx, my) is set to (px + dx, py + dy). The MV is again interpreted as representing 1/4-sample offsets in the luma domain.
5. Store the final value (mx, my) for subsequent use for generating subsequent PMVs (and controlling deblocking filtering, etc.).

[0075] However, the above process is not necessary for all possible embodiments. The approach outlined above assumes that the decoder operates in the ¼ pixel domain from the beginning such that the scaling in the motion vector prediction occurs in the ¼ pixel domain. However, in fact the decoder does not necessarily have to know what the units mean at this stage. Thus is it possible to take an alternative approach whereby no interpretation is placed on the units until later when used to actually generate the image.

[0076] The following describes a decoding process according to embodiments of such an alternative approach. As above, at least one of the motion vector values is differentially encoded, with the motion prediction module 73 being configured to predict the motion vector value by scaling another of said motion vector values. At least one of the motion vector values may be differentially encoded in terms of a delta value, and the motion prediction module 73 may be configured to predict the motion vector value by scaling another of said motion vector values to produce a scaled motion vector value, and then adding the delta value to the scaled motion vector value. However, unlike the above, the motion prediction module 73 is further configured to interpret the motion vector value in said integer pixel units (when the integer mode is being used) without rounding it to integer pixel resolution after the scaling. This is made possible by refraining from placing

an interpretation on the units in the scaling. The motion prediction module 73 may be configured to interpret the motion vector value in said integer pixel units by performing the addition of the delta value to the scaled motion vector value without rounding the scaled motion vector value to integer pixel resolution. The resulting motion vector value is only interpreted in terms of integer or fractional pixel units when used to generate the image. The resulting value is also stored for use in predicting one or more subsequent motion vectors without placing an interpretation on the stored value in terms of either integer or fractional units.

[0077] For example according to such an alternative, the motion vector decoding process may be implemented as follows.

1. Determine how the MV will be predicted at the decoder and whether or not a motion vector delta (MVD) has been sent in the encoded bitstream from the encoder (this may involve some syntax indications).
2. Generate the predicted motion vector (PMV). This creates a pair of integers (px, py). However, don't worry about what this might represent – i.e. do not place any interpretation on the units at this stage.
3. If no MVD is sent, set the MV value (mx, my) to (px, py); otherwise decode the pair of integers (px, py) as (dx, dy), and set the MV (mx, my) to (px + dx, py + dy). If operating in integer-precision mode, interpret the MV as representing whole-integer offsets in the luma domain, i.e. as representing the value (4*mx, 4*my) in 1/4-sample offset units. Otherwise (i.e. when operating in 1/4-sample precision mode), interpret the MV as representing 1/4-sample offsets in the luma domain, i.e. as representing the value (mx, my) in 1/4-sample offset units (N.B. again the luma distinction only matters if the video is in a format such as 4:2:2 or 4:2:0).
4. Store the final value (mx, my) for subsequent use for generating subsequent PMVs (and controlling deblocking filtering, etc.). Again, don't worry about what this might represent.

[0078] Thus that the rounding step 3a described earlier is now eliminated. Only the interpretation of the numbers is modified. This makes the decoding process less complex, because no modification of the MV prediction process is necessary.

[0079] Notre that the rounding discussed above refers to the rounding from ¼ (or fractional) resolution to integer resolution. In present standards there is also a rounding from an even finer precision to the ¼ resolution as part of the motion vector prediction –

and in the alternative embodiment above it is not excluded that such a preliminary rounding is still present.

[0080] There may be some side effects of operating in this manner. One side effect may be that if some pictures use a different mode than others, the scaling used in the MV prediction process for temporal MV prediction won't be correct. However, this would be a very minor effect – especially since it would be rare to have this situation of operating in different modes, and also because temporal MV prediction ordinarily only has a very minor benefit. Another side effect may be that the results of the deblocking filter control may depend on MV values that are interpreted at a different scale factor. However this effect would also be minor, especially since deblocking might be disabled or may not have much of an effect when coding screen content.

[0081] Another possibility would be to disable temporal motion vector prediction whenever using integer motion only. There is already syntax in HEVC that lets the encoder disable the use of that feature. That would be a possible way to avoid needing the decoder to have a special process that operates differently depending on whether the differences are coded as integers or as fractional values. The gain obtained from temporal motion vector prediction may be small (or zero) in these usage cases anyway, so disabling it need not be undesirable.

[0082] It will be appreciated that the above embodiments have been described only by way of example.

[0083] For instance, while the above has been described in terms of blocks, this does not necessarily limit to the divisions called blocks in any particular standard. For example the blocks referred to herein may be the divisions called blocks or macroblocks in the H.26x standards.

[0084] The scope of the disclosure limited to any particular codec or standard and in general the techniques disclosed herein can be implemented either in the context of an existing standard or an update to an existing standard, whether an H.26x standard like H264 or H.265 or any another standard, or may be implemented in a bespoke codec. Further, the scope of the disclosure is not restricted specifically to any particular representation of video samples whether in terms of RGB, YUV or otherwise. Nor is the scope limited to any particular quantization, nor to a DCT transform. E.g. an alternative transform such as a Karhunen-LoeveTransform (KLT) could be used, or no transform may be used. Further, the disclosure is not limited to VoIP communications or communications

over any particular kind of network, but could be used in any network or medium capable of communicating data.

[0085] Where it is said that the motion vector offset is restricted or not restricted to an integer number of pixels, or the like, this may refer to the motion estimation in any one or two of the colour space channels, or the motion estimation in all three colour channels.

[0086] Further, the invention is not limited to selecting between integer pixel and quarter pixel resolution. In general the techniques described herein can be applied to selecting between integer pixel resolution and any fractional pixel resolution, e.g. ½ pixel resolution; or selecting between integer pixel resolution and a plurality of different fractional pixel modes, e.g. selecting between integer, ½ and ¼ pixel modes.

[0087] Further, the scope of the disclosure is not limited to an application in which the encoded video and/or screen capture stream is transmitted over a network, nor in which the streams are live stream. For example in another application, the stream may be stored on a storage device such as an optical disk, hard drive or other magnetic storage, or "flash" memory stick or other electronic memory. Note therefore that a screen sharing stream does not necessarily have to mean live sharing (though that is certainly one option). Alternatively or additionally it could be stored for sharing with one or more other users later, or the captured image data may not be shared but rather just recorded for the user who was using the screen at the time. Generally the screen capture could be any moving image data consisting of captured encoder-side screen content, captured by any suitable means (not necessarily by reading from the screen buffer, though that is one option), to be shared with one or more other users (live or not) or simply recorded for the benefit of the capturing user or for just for archive (perhaps never to actually be viewed again as it may turn out).

[0088] Note also that the codec is not necessarily limited to encoding only screen capture data and video. In embodiments it may also be capable of encoding other types of moving image data, e.g. an animation. Such other types of moving image data may be encoded in the fractional pixel mode or integer pixel mode.

[0089] Further, note that inter frame encoding does not necessarily always have to encode relative to a previous frame, but more generally some codecs may allow encoding relative to a different frame other than the target frame, either preceding or ahead of the target frame (assuming a suitable outgoing buffer).

[0090] Further, as discussed previously, note that motion vectors themselves may be encoded differentially. In this case where it is said that the motion vector as signalled in

the encoded bitstream is restricted to an integer number of pixels, or the like, this means a differentially encoded form of the motion vector is so restricted (e.g. the delta).

[0091] Further, the decoder does not necessarily have to be implemented at an end user terminal, nor output the moving image data for immediate consumption at the receiving terminal. In alternative implementations, the receiving terminal may be an intermediate terminal such as a server running the decoder software, for outputting moving image data to another terminal in decoded or transcoded form, or storing the decoded data for later consumption. Similarly the encoder does not have to be implemented at an end-user terminal, nor encode moving image data originating from the transmitting terminal. In other embodiments the transmitting terminal may for example be an intermediate terminal such as a server running the encoder software, for receiving moving image data in unencoded or alternatively-coded form from another terminal and encoding or transcoding that data for storage at the server or forwarding to a receiving terminal.

[0092] Generally, any of the functions described herein can be implemented using software, firmware, hardware (e.g., fixed logic circuitry), or a combination of these implementations. The terms "module," "functionality," "component" and "logic" as used herein generally represent software, firmware, hardware, or a combination thereof. In the case of a software implementation, the module, functionality, or logic represents program code that performs specified tasks when executed on a processor (e.g. CPU or CPUs). The program code can be stored in one or more computer readable memory devices. The features of the techniques described below are platform-independent, meaning that the techniques may be implemented on a variety of commercial computing platforms having a variety of processors.

[0093] For example, the terminals may include an entity (e.g. software) that causes hardware of the user terminals to perform operations, e.g., processors functional blocks, and so on. For example, the terminals may include a computer-readable medium that may be configured to maintain instructions that cause the user terminals, and more particularly the operating system and associated hardware of the user terminals to perform operations. Thus, the instructions function to configure the operating system and associated hardware to perform the operations and in this way result in transformation of the operating system and associated hardware to perform functions. The instructions may be provided by the computer-readable medium to the terminals through a variety of different configurations.

[0094] One such configuration of a computer-readable medium is signal bearing medium and thus is configured to transmit the instructions (e.g. as a carrier wave) to the

2015204953 05 Sep 2018

computing device, such as via a network. The computer-readable medium may also be configured as a computer-readable storage medium and thus is not a signal bearing medium. Examples of a computer-readable storage medium include a random-access memory (RAM), read-only memory (ROM), an optical disc, flash memory, hard disk memory, and other memory devices that may us magnetic, optical, and other techniques to store instructions and other data.

[0095] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms of implementing the claims.

[0096] Throughout this specification and claims which follow, unless the context requires otherwise, the word "comprise", and variations such as "comprises" and "comprising", will be understood to imply the inclusion of a stated integer or step or group of integers or steps but not the exclusion of any other integer or step or group of integers or steps.

[0097] The reference in this specification to any prior publication (or information derived from it), or to any matter which is known, is not, and should not be taken as an acknowledgment or admission or any form of suggestion that that prior publication (or information derived from it) or known matter forms part of the common general knowledge in the field of endeavour to which this specification relates.

2015204953 05 Sep 2018

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A decoder comprising:

an input receiving moving image data in encoded form, including a plurality of frames in which a region is encoded using motion estimation, the moving image data including a respective set of motion vector values associated with each respective one of the frames or with each respective region within each of the frames, each of the motion vector values of each respective frame or region is included in a motion vector field in an encoded bitstream, the moving image data further including at least one respective indicator in header information associated with each of the respective frames or regions indicating whether the respective motion vector values are encoded at a first resolution or a second resolution, the motion vector field having a reduced size for the frames or the regions having the motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution, and the moving image data including a setting to set whether the first resolution or the second resolution of the motion vector values is being indicated per region or per frame; and

a motion prediction module decoding the frames or regions based on the motion vector values, the decoding comprising:

reading each of the indicators from the header information;

reading the setting and interpreting the motion vector values accordingly;

determining, from the read indicators, whether the motion vector values of the respective frame or region are encoded at the first resolution or the second resolution; and

interpreting the motion vector values in units of fractional pixels for the indicated first resolution or in units of integer pixels responsive for the indicated second resolution.

2. The decoder of claim 1, wherein the moving image data includes two respective indicators associated with each of the frames or regions, the two indicators indicating the resolution of respective motion vectors in two dimensions, and the motion prediction module reading both indicators and interpreting the respective motion vector values accordingly.

2015204953 05 Sep 2018

3. The decoder of claim 1, wherein:

at least one of the frames is divided into multiple regions,

the moving image data includes at least one respective indicator associated with each respective one of the multiple regions to individually indicate whether the motion vector values of the respective region are encoded at the first resolution or the second resolution, and

the motion prediction module reading the indicators to determine whether the motion vector values of each respective region are encoded at the first resolution or the second resolution, and interpreting the respective motion vector values in the units of fractional pixels or integer pixels accordingly.

4. The decoder of claim 3, wherein the multiple regions are slices of an H.26x standard.

5. The decoder of claim 1, wherein the motion prediction module interprets the respective motion vector values in units of fractional pixels as a default if the respective indicators is not present for one of the frames or regions.

6. The decoder of claim 1, wherein at least one of the motion vector values is differentially encoded in terms of a delta value, the motion prediction module predicting the at least one motion vector value by scaling another of the motion vector values, and

the motion prediction module interpreting the at least one motion vector value in the integer pixel units by performing the addition of the delta value to the scaled motion vector value without rounding the scaled motion vector value to integer pixel resolution.

7. The decoder of claim 1, wherein the decoder includes an inverse of a lossless encoding stage preceding the decoding by the motion prediction module.

8. A computer program product embodied on a computer-readable storage hardware memory device that, responsive to execution by a processor performs operations comprising:

receiving moving image data in encoded form, including a plurality of frames in which a region is encoded using motion estimation, the moving image data including a respective set of motion vector values associated with each respective one of the frames or with each

2015204953 05 Sep 2018

respective region within each of the frames, each of the motion vector values of each respective frame or region is included in a motion vector field in an encoded bitstream, the moving image data further including at least one respective indicator in header information associated with each of the respective frames or regions indicating whether the respective motion vector values are encoded at a first resolution or a second resolution, the motion vector field having a reduced size for the frames or the regions having the motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution, and the moving image data including a setting to set whether the first resolution or the second resolution of the motion vector values is being indicated per region or per frame;

reading each of the indicators from the header information;

reading the setting and interpreting the motion vector values accordingly;

determining, from the read indicators, whether the motion vector values of the respective frame or region are encoded at the first resolution or the second resolution;

interpreting the respective motion vector values in units of fractional pixels for the indicated first resolution or in units of integer pixels for the indicated second resolution; and

decoding the frames or regions based on the motion vector values as interpreted in the units of fractional or integer pixels in accordance with the interpreting.

9. The computer program product of claim 8, wherein the moving image data includes two respective indicators associated with each of the frames or regions, the two indicators indicating the resolution of respective motion vectors in two dimensions.

10. The computer program product of claim 8, wherein:

at least one of the frames is divided into multiple regions, and

the moving image data including at least one respective indicator associated with each respective one of the multiple regions to individually indicate whether the motion vector values of the respective region are encoded at the first resolution or the second resolution; and

the computer program product further performing operations including reading the indicators to determine whether the motion vector values of each respective region are encoded at the first resolution or the second resolution, and to interpret the respective motion vector values in the units of fractional pixels or integer pixels accordingly.

2015204953 05 Sep 2018

11. The computer program product of claim 10, the moving image data further including the setting to set whether the resolution of the motion vector values is being indicated per region or per frame, and the computer program product further performing operations including reading the setting and interpreting the motion vector values accordingly.

12. The computer program product of claim 8, the motion prediction module further performing operations including interpreting the respective motion vector values in units of fractional pixels as a default if the respective indicators is not present for one of said frames or regions.

13. A network hardware device comprising: a computer-readable storage hardware memory device having stored thereon a program, that responsive to execution by a processor, generates an encoded bitstream, the encoded bitstream comprising:

a sequence of frames including a plurality of frames in which a region is encoded using motion estimation;

a respective set of motion vector values representing motion vectors of the motion estimation for each respective one of said frames or each respective region within each of said frames, each of the motion vector values of each respective frame or region is included in a motion vector field in the encoded bitstream, the respective set of motion vector values encoded according to a protocol whereby motion vector values encoded at a first resolution represent fractional pixel units and motion vector values encoded at a second resolution represent integer pixel units, wherein according to said protocol, the motion vector field has a reduced size for frames or regions having motion vector values encoded at the second resolution relative to a size of the frames or the regions having the motion vector values encoded at the first resolution; and

a respective indicator in header information associated with each of said respective frames or regions, indicating whether the respective motion vector values of the respective frame or region are encoded at the first resolution or the second resolution as indicated per region or per frame, and enabling a decoder to read the indicator and to interpret the respective motion vector values in units of fractional pixels or integer pixels units by including the indicator in the header information.

2015204953 05 Sep 2018

14. The network hardware device of claim 13, the encoded bitstream including one of the plurality of frames or regions encoded at the first resolution and another of the plurality of frames or regions encoded at the second resolution, the respective indicator indicating the resolution individually for each of said plurality of frames or regions.

15. The network hardware device of claim 13, wherein:

at least one of the frames is divided into multiple regions; and

the encoded bitstream includes at least one respective indicator associated with each respective one of the multiple regions to individually indicate whether the motion vector values of the respective region are encoded at the first resolution or the second resolution.

16. The network hardware device of claim 15, wherein the regions are slices of an H.26x standard.

17. The network hardware device of claim 15, wherein the encoded bitstream further including includes a setting to set whether the resolution of the motion vector values is being indicated per region or per frame.

18. The network hardware device of claim 13, wherein the respective motion vector values included in the encoded bitstream are further encoded according to a lossless encoding technique.

19. The network hardware device of claim 13, wherein the respective motion vector values are interpreted in units of fractional pixels as a default if the respective indicators is not present for one of the frames or regions.

20. The network hardware device of claim 13, wherein the encoded bitstream comprises a setting to set whether the resolution of the motion vector values is being indicated per region or per frame, and the setting being readable by which to interpret the motion vector values accordingly.

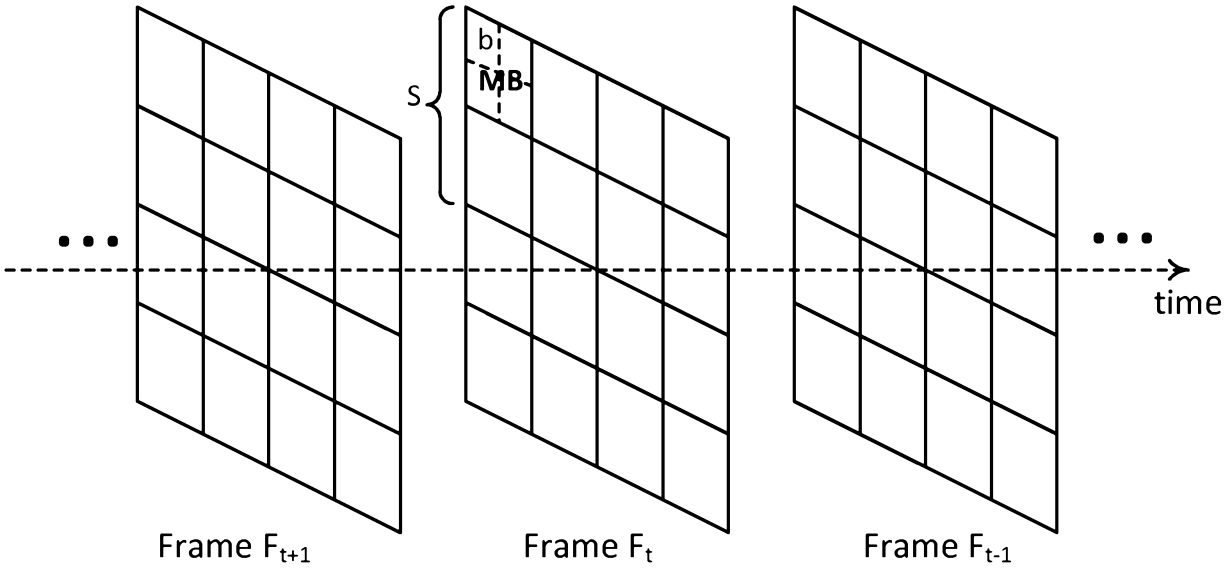

Figure 1

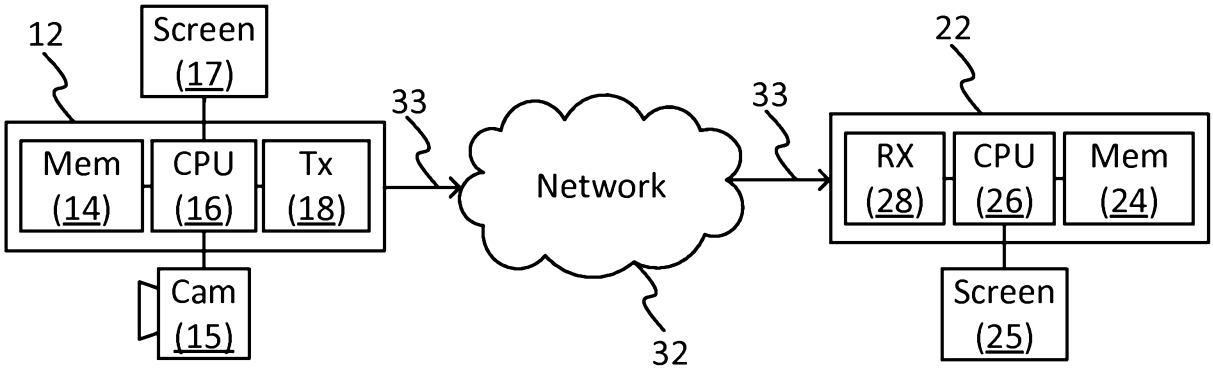

Figure 2

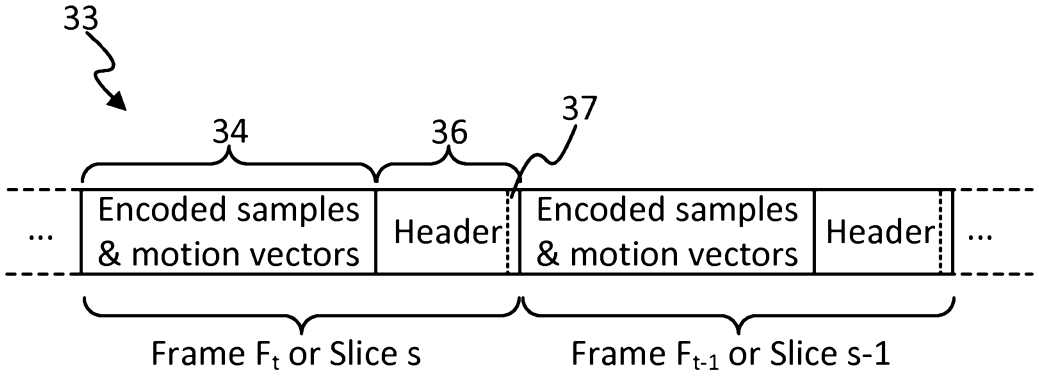

Figure 3

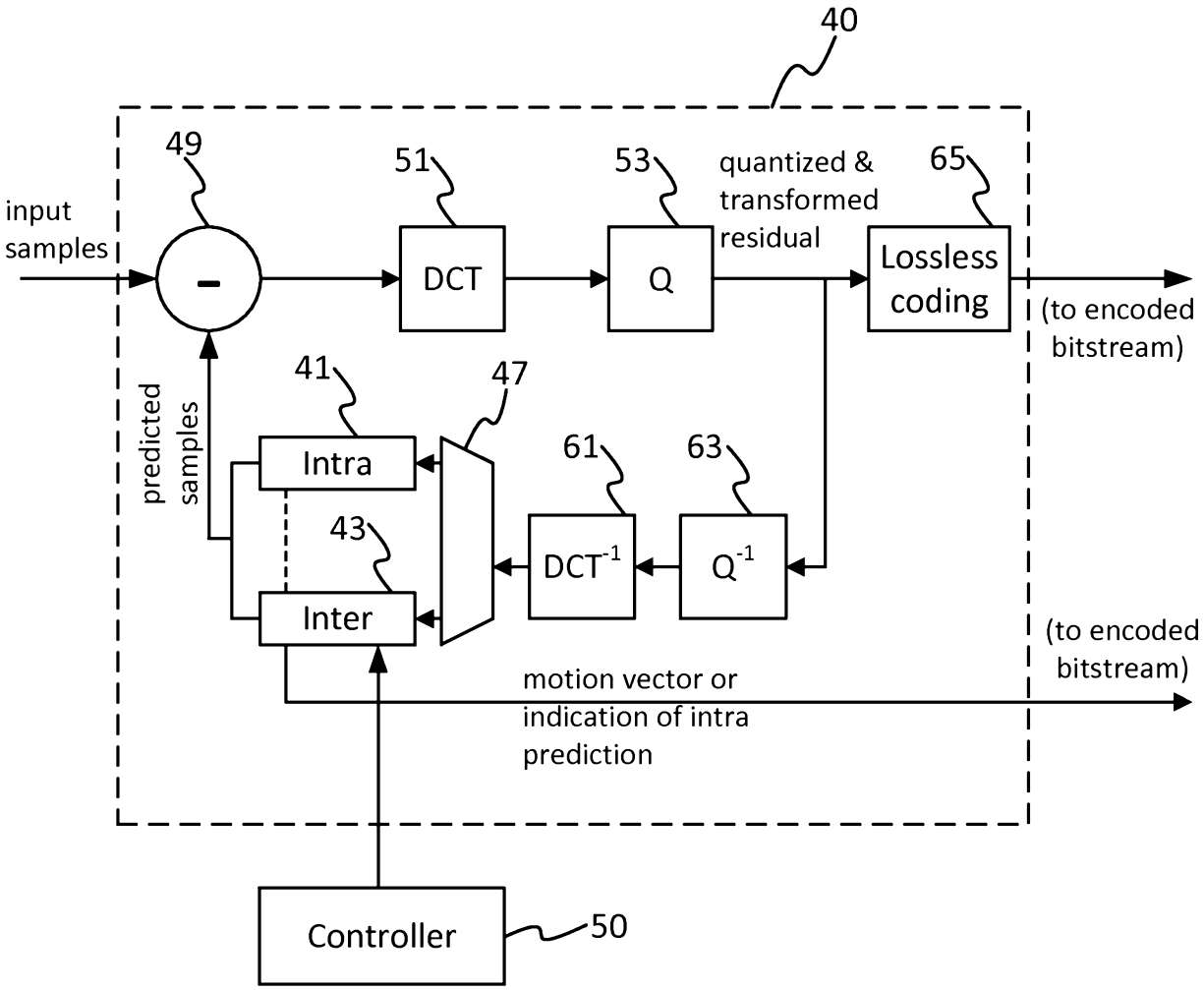

Figure 4

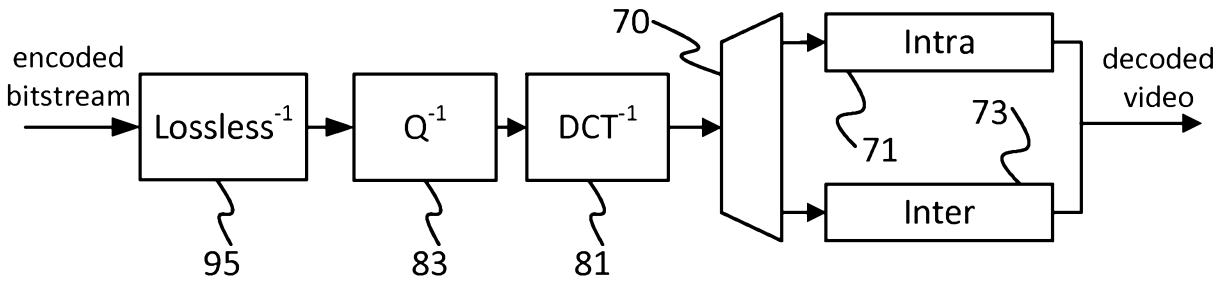

Figure 5

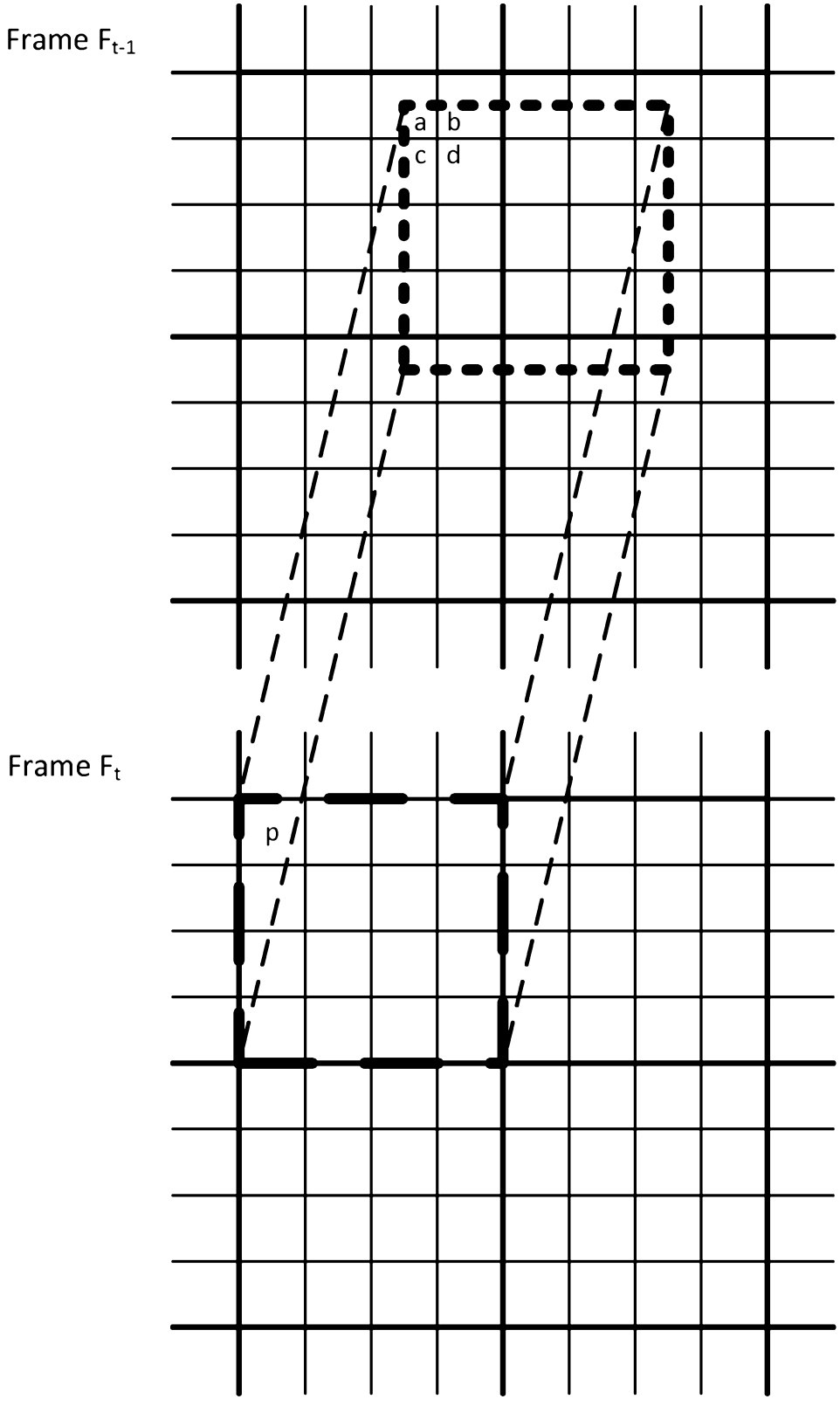

Figure 6

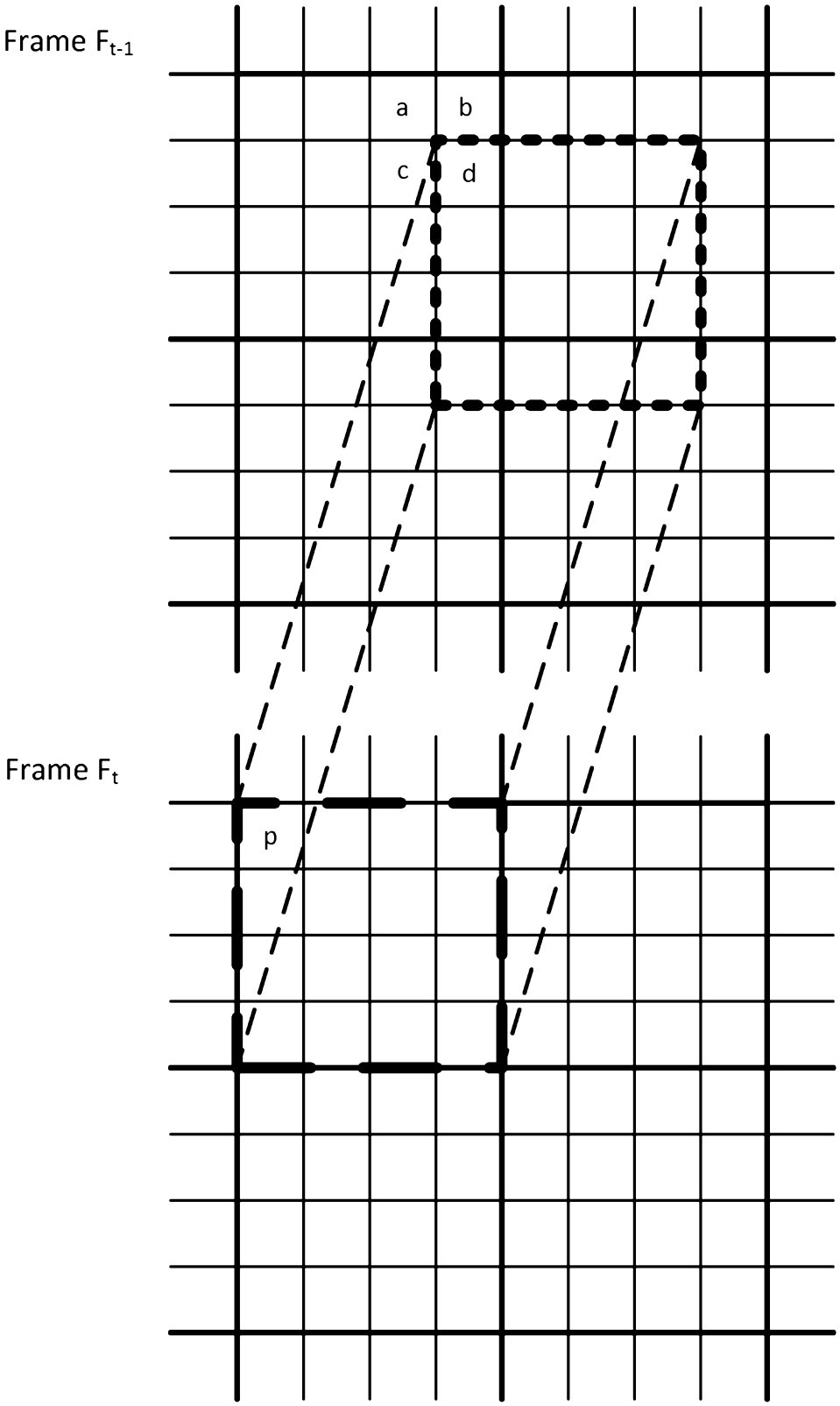

Figure 7