

METHOD AND APPARATUS FOR MOTION VECTOR ASSIGNMENT**Claim of Priority under 35 U.S.C. §119**

[0001] The present Application for Patent claims priority to Provisional Application No. 60/590,110, and assigned to the assignee hereof and hereby expressly incorporated by reference herein.

Reference to Co-Pending Applications for Patent

[0002] The present Application for Patent is related to the following co-pending U.S. Patent Applications:

[0003] “Method and Apparatus for Motion Compensated Frame Rate Up Conversion for Block-Based Low Bit-Rate Video”, having U.S. Application Serial No. 11/122,678, filed concurrently herewith, assigned to the assignee hereof, and expressly incorporated by reference herein.

BACKGROUND**Field**

[0004] The embodiments described herein relate generally to video image processing, and more particularly, to a method and apparatus for motion vector assignment.

Background

[0005] Low bit rate video compression is very important in many multimedia applications such as wireless video streaming and video telephony, due to the limited bandwidth resources and the variability of available bandwidth. Bandwidth adaptation video coding at low bit-rate can be accomplished by reducing the temporal resolution. In other words, instead of compressing and sending a thirty (30) frame per second (fps) bit-stream, the temporal resolution can be halved to 15 fps to reduce the transmission bit-rate. One typical approach for reducing the bandwidth is through the extraction and removal of redundant information from the encoded signal, such as removing either a portion of or the complete frame. However, the consequence of reducing temporal resolution is the introduction of temporal domain artifacts such as motion jerkiness that significantly degrades the visual quality of the decoded video.

[0006] To display the full frame rate at the receiver side, a recovery mechanism, called frame rate up conversion (FRUC), is needed to re-generate the skipped frames and to reduce temporal artifacts. Generally, FRUC is the process of video interpolation at the video decoder to increase the perceived frame rate of the reconstructed video. Many FRUC algorithms have been proposed, and can be classified into two general categories. The first category interpolates the missing frame by using a combination of received video frames without taking object motion into account. Frame repetition and frame averaging methods fit into this class. The drawbacks of these methods include the production of motion jerkiness, "ghost" images and blurring of moving objects when there is motion involved. The second category is more advanced as compared to the first category, and utilizes motion information transmitted by the encoder in a process referred to as motion compensated (frame) interpolation (MCI).

[0007] In conventional motion compensated predictive coding, each video frame is first partitioned into square blocks of picture elements (pels); such as blocks of 8 pels by 8 pels. Each block is coded, in turn, and the developed encoded sequence is transmitted over a communications channel to a decoder. During the encoding process, a determination is made as to whether or not the pels of the block have changed significantly compared with a collocated block of a previous frame. If not, an indication is sent that signifies to the decoder that it needs to merely repeat the pels of that block from the previous frame to obtain the pels for the current block. If the pels have changed since the previous frame, an attempt is made to determine the best estimate of motion that is occurring in the block. This is frequently done by a block matching motion estimation technique, where the pels of the current block are successively compared with various small shifts of the corresponding block in the previous frame. The shift that provides the best match is deemed to be the "best estimate" of the displacement in the block's image between frames, and the magnitude and direction of this shift, referred to as a "motion vector," is selected to be sent to the decoder. Also, once the best match is found, the pels of the current block are compared with those of the "best" shifted block from the previous frame to see if there is a significant difference. If not, instead of sending the motion vector, an indication is sent to the decoder so that the pels of the shifted block from the previous frame will be repeated for the pels for the current shifted block. Such blocks are said to have been successfully motion compensated. However, if there is a significant difference between the two

blocks, the difference is encoded and sent to the decoder so that the pels of the current block may be more accurately recovered.

[0008] There have been some suggestions for improving the quality of the repeated frames in order to make them more faithfully resemble the original. Thus, instead of simply repeating the pels from the previous frame as described above in conventional motion compensated predictive coding, in MCI motion vectors are used to laterally displace a block by the appropriate amount prior to display. In other words, this method creates the missing block of pels by averaging over the immediately previous and following blocks of pels that are available to the decoder. However, experimental results show that when the images of successive blocks do not represent translational motion, the reproduced image may be worse than with frame repeating. Although it has been observed that this degradation is caused by a relatively few pels that do not conform to the assumption of translational motion, putting these pels in the wrong place creates highly visible artifacts.

[0009] As illustrated in prior art FIG. 1, in MCI, a missing frame 108 is interpolated based on a reconstructed current frame 102, a stored previous frame 104, and a set of transmitted motion vectors 106. Reconstructed current frame 102 is composed of a set of non-overlapped blocks 150, 152, 154 and 156 associated with set of transmitted motion vectors 106 pointing to corresponding blocks in stored previous frame 104. Interpolated frame 108 can be constructed in either a linear combination of corresponding pixels in current and previous frames; or nonlinear operation such as a median operation.

[0010] Although MCI is more advanced, block-based MCI introduces overlapped regions, which are areas having multiple motion trajectories passing through it; and hole regions, which are areas having no motion trajectory passing through it, both of which are unwanted, in interpolated frames. As illustrated in FIG. 2, an interpolated frame 202 contains an overlapped area 206 and a hole area 204. The main reasons for the generation of these two types of unwanted areas are:

[0011] 1. Moving objects are not under a rigid translational motion model.

[0012] 2. The transmitted motion vectors used in MCI may not point to the true motion trajectories due to the block-based fast motion search algorithms utilized on the encoder side.

[0013] 3. The covered and uncovered background in the current frame and previous frames.

[0014] The interpolation of overlapped and hole regions is a major technical challenge in conventional block-based motion compensated approaches. Median blurring and spatial interpolation techniques have been proposed to fill these overlapped and hole regions. However, the drawbacks of these methods are the introduction of the blurring and blocking artifacts, and also an increase in the complexity of interpolation operations.

[0015] Proper motion vector assignment is important for motion compensated FRUC as it is needed for the interpolated frames in motion compensated FRUC. In addition, proper motion vector assignment is useful in any application where proper motion compensation based on motion vector assignment is needed. Accordingly, there is a need to overcome the issues noted above.

SUMMARY OF THE EMBODIMENTS

[0016] The embodiments described herein provide a motion vector assignment process for frames that are to be interpolated having holes and overlapped areas.

[0017] In one embodiment, a method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, is described. The method includes partitioning the video frame to be interpreted into a plurality of areas; determining a number of motion vectors passing through one area based on the content maps of the first reference frame and the second reference frame; and assigning a motion vector to the one area based on the number of motion vectors passing through the one area.

[0018] In another embodiment, a computer readable medium having instructions stored thereon for causing a computer to execute a method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, is disclosed. The method includes partitioning the video frame to be interpreted into a plurality of areas; determining a number of motion vectors passing through one area based on the content maps of the first reference frame and the second reference frame; and assigning a motion vector to the one area based on the number of motion vectors passing through the one area.

[0019] In yet another embodiment, an apparatus for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, is disclosed. The apparatus includes means for partitioning the video frame to be interpreted into a plurality of areas; means for determining a number of motion vectors passing through one area based on the content maps of the first reference frame and the second reference frame; and means for assigning a motion vector to the one area based on the number of motion vectors passing through the one area.

[0020] In still yet another embodiment, at least one processor configured to implement a method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, is disclosed. The method includes partitioning the video frame to be interpreted into a plurality of areas; determining a number of motion vectors passing through one area based on content maps of the first reference frame and the

second reference frame; and assigning a motion vector to the one area based on the number of motion vectors passing through the one area.

[0021] Other objects, features and advantages will become apparent to those skilled in the art from the following detailed description. It is to be understood, however, that the detailed description and specific examples, while indicating exemplary embodiments, are given by way of illustration and not limitation. Many changes and modifications within the scope of the following description may be made without departing from the spirit thereof, and the description should be understood to include all such variations.

BRIEF DESCRIPTION OF THE DRAWINGS

- [0022] The embodiments described herein may be more readily understood by referring to the accompanying drawings in which:
- [0023] FIG. 1 is a figure illustrating the construction of an interpolated frame using motion compensation interpolation (MCI);
- [0024] FIG. 2 is a figure illustrating overlapping and hole regions that may be encountered in an interpolated frame during MCI;
- [0025] FIG. 3 is a block diagram for a FRUC system for performing motion vector assignment;
- [0026] FIG. 4 is a block diagram of a motion vector extraction subsystem of a motion vector a system;
- [0027] FIG. 5 is a flow diagram illustrating an overview of a method for motion vector assignment for overlapped and hole block regions;
- [0028] FIG. 6 is a flow diagram illustrating an operation of the FRUC system for motion vector assignment for overlapped and hole block regions;
- [0029] FIG. 7 is a figure illustrating the various content classes assigned to the graphic elements inside a video frame; and,
- [0030] FIG. 8 is a flow diagram illustrating an operation of the FRUC system for motion vector assignment for overlapped and hole pixel regions; and,
- [0031] FIG. 9 is a block diagram of an access terminal and an access point of a wireless system.
- [0032] Like numerals refer to like parts throughout the several views of the drawings.

DETAILED DESCRIPTION

[0033] The embodiments described herein provide motion vector assignment methods and apparatus that supports variable block-size-based, block-based, and pixel-based variations in motion vector assignment. The motion vector assignment methods and apparatus described herein assists in the creation of interpolated frames, also referred to as "F frames," with higher quality in, for example, a process such as a frame rate up conversion (FRUC) process. Supplementary information contained in the decoded video data are utilized as inputs to the motion vector assignment process. Supplementary information may include, for example, a description of the type of graphic elements that make up each frame, referred to as "content maps," as well as a description of the complexity and movement of the elements, referred to as "complexity maps" and "activity maps," respectively. The motion vector assignment described herein addresses such problems encountered during the creation of interpolated frames as overlapped and hole areas, both of which are unwanted in interpolated frames.

[0034] FIG. 3 illustrates a FRUC system 300 that is able to perform motion vector assignment for F-frames, also referred to as to-be-interpolated frames, where FRUC system 300 receives an incoming bit stream 302 and uses a binary decoder module 306 to extract: (1) a set of motion vectors 308, which is placed in a stored motion vector buffer 316; and, (2) a residue 310. Residue 310 is input into an inverse quantization/inverse transform module 312, the result of which is combined with a previous frame stored in a stored previous frame buffer 318 to generate a current frame. The current frame is stored in a current frame buffer 320. A sub-portion 350 of FRUC system 300, including stored motion vector buffer 316, stored frame buffer 318, and current frame buffer 320, contains the specific functional modules for performing motion vector assignment in accordance with one exemplary embodiment, as described herein. Specifically, sub-portion 350 include a motion vector processor 322, a seeded bidirectional motion estimation module 324, a second motion vector processor 326, a mode decision module 328, a frame rate up converter (FRUC) module 330, and a post-processing unit 332. The operation of FRUC system 300 is further described below in connection with the description of the figures. In addition, one exemplary motion vector extrapolation and content classification system that may be used with the system described herein is described in co-pending application No. _____, entitled "Method and Apparatus for Frame Rate Up Conversion". [Fang Shi, Viji Raveendran. June 15,

2004]. Further, one exemplary motion vector smoothing process that may be used with the system described herein is described in co-pending application No. _____, entitled "Method and Apparatus for Motion Vector Processing" [Fang Shi, Viji Raveendran]. Co-pending application No. _____, entitled "Method and Apparatus for Motion Compensated Frame Rate Up Conversion." [040432], also provides other exemplary embodiments of the other components of the FRUC system 300 not specifically described herein.

[0035] It should be noted that the functional modules shown in FIG. 3, in one embodiment, correspond to specific modules that may be implemented using one or more software algorithms. The operation of the algorithms is described at a high-level with sufficient detail to allow those of ordinary skill in the art to implement them using a combination of hardware and software approaches. For example, the components described herein may be implemented as software executed on a general-purpose processor; as "hardwired" circuitry in an Application Specific Integrated Circuit (ASIC); or any combination thereof. It should be noted that various other approaches to the implementation of the modules described herein may be employed and should be within the realm of those of ordinary skill of the art who practice in the vast field of image and video processing.

[0036] Further, the inventive concepts described herein may be used in decoder/encoder systems that are compliant with H.26x-standards as promulgated by the International Telecommunications Union, Telecommunications Standardization Sector (ITU-T); or with MPEGx-standards as promulgated by the Moving Picture Experts Group, a working group of the International Standardization Organization/International Electrotechnical Commission, Joint Technical Committee 1 (ISO/IEC JTC1). The ITU-T video coding standards are called recommendations, and they are denoted with H.26x (e.g., H.261, H.262, H.263 and H.264). The ISO/IEC standards are denoted with MPEG-x (e.g., MPEG-1, MPEG-2 and MPEG-4). For example, multiple reference frames and variable block size are special features required for the H264 standard. In other embodiments, the decoder/encoder systems in which the inventive concepts described herein are used may be proprietary.

[0037] In one embodiment, FRUC system 300 may be configured based on different complexity requirements. For example, a high complexity configuration may include multiple reference frame; variable block sizes; previous reference frame motion vector

extrapolation with motion acceleration models; and, motion estimation assisted double motion field smoothing. In contrast, a low complexity configuration may only include a single reference frame; fixed block sizes; and motion compensated interpolation with motion field smoothing. Other configurations are also valid for different applications.

[0038] System 300 receives input from a plurality of data storage units in a data subsystem 400 used to store and extract information about the video frames used in the processing. As illustrated in FIG. 4, these storage units include including a multiple previous frames' content maps storage unit (MULTIPLE F_PREV CONTENT MAP) 402; a multiple previous frames' extrapolated motion fields storage unit (EXTRAPOLATED MULTIPLE F_PREV MOTION FIELDS) 404; a single previous frame content map storage unit (F_PREV CONTENT MAP) 406; and a single previous frame extrapolated motion field storage unit (EXTRAPOLATED F_PREV MOTION FIELD) 408. A multiple reference frame mode controller module 414 couples the appropriate storage units to the next stage of input depending on whether system 300 is operating in a mode that utilizes multiple reference frames. Specifically, when system 300 utilizes multiple previous reference frames, the inputs are taken from multiple previous frames' content maps storage unit 402 and multiple previous frames' extrapolated motion fields storage unit 404. For the single previous reference frame mode of operation, the inputs are taken from single previous frame content map storage unit 406 and single previous frame extrapolated motion field storage unit 408.

[0039] Although the figure illustrates the use of two different sets of storage units for storing content maps and motion fields—one set for where multiple reference frames are used (i.e., multiple previous frames' content maps storage unit 402 and multiple previous frames' extrapolated motion fields storage unit 404) and another for where a single reference frame is used (i.e., single previous frame content maps storage unit 406 and single previous frame extrapolated motion field storage unit 408), it should be noted that other configurations are possible. For example, the functionality of the two different content map storage units may be combined such that one storage unit may be used to store the content maps for multiple frames or a single content map for a single frame. Further, the storage units may also store data for the current frame as well.

[0040] The output from multiple reference frame mode controller module 414 as well as a current frame motion field storage unit (F_CURR MOTION FIELD) 410 and a current frame content map storage unit (F_CURR CONTENT MAP) 412 are fed into a

motion vector extrapolation unit 416. Motion vector extrapolation unit 416 extracts the motion vectors as well as the supplemental information used for motion vector processing in system 300, as described herein. Thus, the input motion vectors utilized in system 300 may be from the current frame, or may be from both the current frame and one or more of the previously decoded frames. In addition, the other input to system 300 is side-band information from the decoded frame data, which may include, but is not limited to, the regions of interest, variations in texture information, and variations in luminance background values. This information may provide guidance for motion vector classification and adaptive smoothing algorithms, as further discussed below.

[0041] FIG. 5 illustrates an overview of an exemplary motion vector assignment process 500, which starts with content maps and motion vectors for the reference frames used to interpolate the F frame being obtained in step 502. In one embodiment, where the process is being performed on a decoder on a client, the content maps and motion vectors have been previously transmitted from a server, as discussed above. In another embodiment, the content map or motion vectors may be generated on the client.

[0042] In step 504, each block in F frame (i.e., the frame to be interpolated), also referred to as "F frame block," "F frame macroblock," or "F_MB," are classified as either hole areas or overlapped areas. As further described herein, to determine if each block in the F frame contains an overlapped or hole area, the F frame is first divided into a series of non-overlapping blocks. It is then determined whether any motion vectors starting and ending on the reference frames between which the F frame is located passes through each non-overlapping block in the F frame. As discussed previously, overlapped areas are regions having multiple motion trajectories (i.e., multiple motion vectors) passing through it, while hole areas are regions having no motion trajectory passing through it (i.e., no motion vectors). Thus, if a block in the F frame has multiple motion vectors passing through it, it will be classified as containing an overlapped area, also referred to herein as an "overlapped area block" or "overlapped area." Further, the multiple motion vectors passing through a particular block, which causes the block to be an overlapped area block, will be referred to as a "set of overlapping motion vectors" for the block. This set of overlapping motion vectors will be subject to modification where certain motion vectors may be eliminated from the set of overlapping motion vectors, as further described below, and the modified set of

motion vectors are referred to herein as a "processed set of overlapped motion vectors." In another embodiment, the set of overlapping motion vectors only contain motion vectors that meet certain criterion, and thus will be equal to the processed set of overlapped motion vectors. If a block does not have any motion vectors passing through it, it will be classified as a hole area, also referred to herein as a "hole area block" or "hole area." In one embodiment, a block having a single motion vector passing through it will not be classified as having a hole area. In another embodiment, these types of blocks will be classified as an overlapped area. Once blocks have been classified, the process continues with the motion assignment process, which depends on how a block is classified.

[0043] In step 506 it is determined if a block has been classified as a hole area. If so, then operation continues with step 514, where a motion vector assignment process for hole areas is performed. Otherwise, operation continues with step 508, where it is determined if only a single motion vector passes through the block in the F frame. If so, operation continues with step 512, where a standard motion compensation process occurs based on the single motion vector. If the block has been classified as an overlapped area, then operation continues with step 510, where a motion vector assignment process is performed for overlapped areas is performed.

[0044] FIG. 6 illustrates an exemplary motion vector assignment process, where the general motion vector assignment process described with reference to FIG. 5 is applied to the operation of FRUC system 300 for performing motion vector assignment for overlapped and hole blocks, where the system begins by first partitioning an F frame--in this case, a frame to be interpreted, into non-overlapping macroblocks (MBs) and assigning the hole area or overlapped area classifications to these blocks in step 602. In a fixed block size system, a frame must be divided into blocks of one fixed size. As described herein, a macroblock (MB) refers to a 16x16-sized block. However, the description herein may apply to other sized blocks. Further, in a system that supports variable block sizes, a frame may be divided into blocks that have different sizes and shapes. For video codecs that support variable block sizes, system 300 can also perform motion vector assignment for the motion vectors associated with these blocks. One such standard is the H.264 standard, as mentioned above, which supports block sizes of 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4.

[0045] Once the F frame has been partitioned, the blocks are classified as hole or overlapped areas in a manner as discussed above, where all the motion vectors that belong to an overlapping area block are grouped into respective sets, operation continues with block 604, where FRUC system 300 determines if content maps exist for the reference frames.

[0046] It is important for the content in a frame to be classified as part of the motion vector assignment process described herein. Based on the received video stream metadata (i.e., transmitted motion vectors) and the decoded data (i.e., reconstructed frame pixel values), the content in a frame can be classified into the following class types:

- [0047] 1. static background (SB);
- [0048] 2. moving object (MO);
- [0049] 3. appearing object (AO);
- [0050] 4. disappearing object (DO); and,
- [0051] 5. edges (EDGE).

[0052] Thus, the class type of the region of the frame at which the current motion vector is pointing--i.e., the macroblock or block, is analyzed and will affect the processing of the frames that are to be interpolated. The introduction of EDGE class to the content classification adds an additional class of content classification and provides an improvement in the FRUC process, as described herein.

[0053] FIG. 7 provides an illustration of the different classes of pixels or regions, including a moving object (MO) 708, an appearing object (AO) 704, a disappearing object (DO) 710, a static background (SB) 702 and an EDGE 706 classes for MCI, where a set of arrows 712 denotes the motion trajectory of the pixels in the three illustrated frames: Frame (t-1), Frame (t) (F Frame) and Frame (t+1). Specifically, in the context of MCI, each pixel or region inside each video frame can be classified into one of the above-listed five classes and an associated motion vector may be processed in a particular fashion based on a comparison of the change (if any) of class type information, as further described below. In addition, the above-mentioned five content classifications can be group into three less-restricted classes when the differences between the SB, AO and DO classes are minor:

- [0054] 1. SB 702, AO 704, DO 710;
- [0055] 2. MO 708; and,

[0056] 3. EDGE 706.

[0057] In one embodiment, two different approaches are used to perform the classification of DO 710, SB 702, AO 704 and MO 708 content, each based on different computational complexities. In the low-complexity approach, for example, the following formulas may be used to classify content:

[0058] $Q_c = \text{abs}(F_c[y_n][x_n] - F_p[y_n][x_n]);$

[0059] $Q_p = \text{abs}(F_p[y_n][x_n] - F_{pp}[y_n][x_n]);$

[0060] $Q_c = (Q_c > \text{threshold});$ and,

[0061] $Q_p = (Q_p > \text{threshold});$

[0062] where:

[0063] y_n and x_n are the y and x coordination positions of the pixel;

[0064] F_c is the current frame's pixel value;

[0065] F_p is the previous frame's pixel value;

[0066] F_{pp} is the previous-previous frame pixel value;

[0067] Q_c is the absolute pixel value difference between collocated pixels (located at $[y_n][x_n]$) in current- and previous frames; and,

[0068] Q_p is the absolute pixel value difference between collocated pixels (located at $[y_n][x_n]$) in previous- and previous-previous frames;

[0069] and:

[0070] if ($Q_c \ \&\& \ Q_p$) then classify as moving object;

[0071] else if ($!Q_c \ \&\& \ !Q_p$) then classify as stationary background;

[0072] else if ($Q_c \ \&\& \ !Q_p$) then classify as disappearing object;

[0073] else if ($!Q_c \ \&\& \ Q_p$) then classify as appearing object.

[0074] In the high-complexity approach, for example, classification is based on object segmentation and morphological operations, with the content classification being performed by tracing the motion of the segmented object. Thus:

[0075] 1. perform object segmentation on the motion field;

[0076] 2. trace the motion of the segmented object (e.g., by morphological operations); and,

[0077] 3. mark the object as SB, AO, DO, and MO, respectively.

[0078] In step 606, after content maps are determined to exist for the reference frames, the FRUC system will eliminate, for each overlapped area macroblock in the F frame, any motion vectors from the set of overlapped motion vectors belonging to the overlapped area block that starts from or ends with different content classes.

Specifically, by using the content map of each reference frame to determine the content type of each macroblock in that reference frame, where each macroblock is classified to be one of the MO, SB, AO, DO and EDGE classes (or into three classes where differences between the SB, AO and DO classes are minor) as described above, it can be determined if any motion vector in the set of overlapping motion vectors (i.e., previously having been defined as all motion vectors passing through an overlapped area block) start and end with macroblocks having different content classes. For example, a motion vector having its starting point touching a macroblock in Frame (T+1) classified as a MO macroblock but its ending point touching a macroblock in Frame (T-1) classified as a SB macroblock would be such a motion vector. In one embodiment, any motion vector is eliminated from the set of overlapped motion vectors if that motion vector's starting and ending points belong to different content classes. As defined previously, the motion vectors that have not been eliminated are the processed set of motion vectors.

[0079] Once the motion vectors having starting and ending points associated with macroblocks classified in different classes have been removed from the set of overlapped motion vectors and the processed set of motion vectors created, operation continues with decision block 608, where it is determined if the macroblock to be assigned a motion vector is an overlapped area macroblock.

Motion Vector Assignment for Overlapped MB

[0080] In step 608, it is determined whether the processed set of overlapped motion vectors for each macroblock contains more than one overlapped motion vector. If so, then in one embodiment operation continues with step 616, where the motion vector(s) passing through the center pixel is used in the assignment of the motion vector to the macroblock. In another embodiment, as noted in step 618, the assignment is based on a most covered area determination. In yet another embodiment, as noted in step 620, a weighted motion vector is assigned, the weighted motion vector being created based on the coverage in the overlapped area macroblock. In still yet another embodiment, as noted in step 622, a median motion vector is assigned. Each of these embodiments are described below.

Assign MV based on the center pixel's MV:

- [0081] In one embodiment, the motion vector assignment is based on determining the motion vector that passes through the center pixel of the macroblock. The sequence including the following steps:
- [0082] 1. start with the current MB($x/\text{block_size}$, $y/\text{block_size}$) in the current frame, where (x,y) is the center pixel location of the current macro block;
- [0083] 2. calculate the current motion vector (dx , dy) of the MB(x,y)
- [0084] 3. scale the motion vector proportionally, for 1:2 FRUC case, where the F frame is exactly between the current frame and the previous reference frame, it is ($dx/2$, $dy/2$)
- [0085] 4. the MB in the F-frame, MB_f($(x+dx/2)/\text{block_size}$, $(y+dy/2)/\text{block_size}$) will be assigned the MV($dx/2$, $dy/2$)
- [0086] 5. if there are more than one MVs assigned to the same MB in the F-frame, an averaging or median operation will be performed to obtain the final seed MV.

Assign MV based on the MB that covers most of the area:

- [0087] In another embodiment, the motion vector assignment is based on determining the motion vector with an associated macroblock that covers the most area of the F frame's macroblock. The steps are as follows:
- [0088] 1. current MB($x/\text{block_size}$, $y/\text{block_size}$) in the current frame, where (x,y) is its center pixel location
- [0089] 2. current MV(dx , dy) of the MB(x,y)
- [0090] 3. scale the MV proportionally, for 1:2 FRUC case, it is ($dx/2$, $dy/2$)
- [0091] 4. the MV(dx , dy) passes thru the F-frame, the intersection location in F-frame is ($x+dx/2$, $y+dy/2$)
- [0092] 5. find the up to four MBs in F-frame overlay with the MB centered around ($x+dx/2$, $y+dy/2$)
- [0093] a. F_MB_1 (x_1 , y_1); where $x_1 = ((\text{int}) (x+dx/2)/\text{block_size}) * \text{block_size} + \text{block_size}/2$; $y_1 = ((\text{int}) (y+dy/2)/\text{block_size}) * \text{block_size} + \text{block_size}/2$;
- [0094] b. F_MB_2 (x_2 , y_2); where $x_2 = x_1 + \text{block_size}$; $y_2 = y_1$;
- [0095] c. F_MB_3 (x_3 , y_3); where $x_3 = x_1 + \text{block_size}$; $y_3 = y_1 + \text{block_size}$;
- [0096] d. F_MB_4 (x_4 , y_4); where $x_4 = x_1$; $y_4 = y_1 + \text{block_size}$;
- [0097] 6. find up to four areas within the four F_MB's covered by the MB centered around ($x+dx/2$, $y+dy/2$), and associated all the calculated areas with MV ($dx/2$, $dy/2$)

- [0098] a. $\text{Area_MB_1} = (\text{block_size} - (x + dx/2 - x_1)) * (\text{block_size} - (y + dy/2 - y_1))$
- [0099] b. $\text{Area_MB_2} = (\text{block_size} - (x + dx/2 - x_2)) * (\text{block_size} - (y + dy/2 - y_2))$
- [0100] c. $\text{Area_MB_3} = (\text{block_size} - (x + dx/2 - x_3)) * (\text{block_size} - (y + dy/2 - y_3))$
- [0101] d. $\text{Area_MB_4} = (\text{block_size} - (x + dx/2 - x_4)) * (\text{block_size} - (y + dy/2 - y_4))$
- [0102] 7. for each F_MB in the F-frame, pick the MV which gives the largest area.

Weighted average MV assignment:

- [0103] In still yet another embodiment, the motion vector assignment is based on a weighted average of the macroblocks that cover the macroblock. The steps are as follows:
- [0104] 1. current MB($x/\text{block_size}$, $y/\text{block_size}$) in the current frame, where (x, y) is its center pixel location
- [0105] 2. current MV(dx , dy) of the MB(x, y)
- [0106] 3. scale the MV proportionally, for 1:2 FRUC case, it is ($dx/2$, $dy/2$)
- [0107] 4. the MV(dx , dy) passes thru the F-frame, the intersection location in F-frame is ($x + dx/2$, $y + dy/2$)
- [0108] 5. find the up to four MBs in F-frame overlay b with the MB centered around ($x + dx/2$, $y + dy/2$)
- [0109] a. F_MB_1 (x_1 , y_1); $x_1 = ((\text{int}) (x + dx/2) / \text{block_size}) * \text{block_size} + \text{block_size} / 2$;
 $y_1 = ((\text{int}) (y + dy/2) / \text{block_size}) * \text{block_size} + \text{block_size} / 2$;
- [0110] b. F_MB_2 (x_2 , y_2); $x_2 = x_1 + \text{block_size}$; $y_2 = y_1$;
- [0111] c. F_MB_3 (x_3 , y_3); $x_3 = x_1 + \text{block_size}$; $y_3 = y_1 + \text{block_size}$;
- [0112] d. F_MB_4 (x_4 , y_4); $x_4 = x_1$; $y_4 = y_1 + \text{block_size}$;
- [0113] 6. find up to four areas within the four F_MB_s covered by the MB centered around ($x + dx/2$, $y + dy/2$), and associated all the calculated areas with MV ($dx/2$, $dy/2$)
- [0114] a. $\text{Area_MB_1} = (\text{block_size} - (x + dx/2 - x_1)) * (\text{block_size} - (y + dy/2 - y_1))$
- [0115] b. $\text{Area_MB_2} = (\text{block_size} - (x + dx/2 - x_2)) * (\text{block_size} - (y + dy/2 - y_2))$
- [0116] c. $\text{Area_MB_3} = (\text{block_size} - (x + dx/2 - x_3)) * (\text{block_size} - (y + dy/2 - y_3))$
- [0117] d. $\text{Area_MB_4} = (\text{block_size} - (x + dx/2 - x_4)) * (\text{block_size} - (y + dy/2 - y_4))$
- [0118] 7. if N MVs pass thru a F_MB, then assign the weighted MV to the F_MB
- [0119] $\text{MV} = w_1 * \text{MV}_1 + w_2 * \text{MV}_2 + \dots + w_N * \text{MV}_N$
- [0120] where the w_i is the weighting factor:
- [0121] $w_i = \text{area}_i / \text{block_size} / \text{block_size}$;

[00122] In step 628, where it has been previously determined in step 608 that the F frame macroblock does not have any overlapped motion vectors, the system determines if the F frame macroblock has a single motion vector passing through it (i.e., whether there is only one motion vector in the processed set of motion vectors). If so, then operation continues with step 630, where a normal motion vector assignment is performed. Otherwise, the F frame macroblock is determined to not have any motion vectors passing through it and operation continues with step 622, where in one embodiment a motion vector with a zero value is assigned to the F frame macroblock. In another embodiment, as noted in step 624, the F frame macroblock is assigned a motion vector that is an average of the available motion vectors neighboring the macroblock. In yet another embodiment, as noted in step 626, the F frame macroblock is assigned a motion vector that is the median of the available motion vectors neighboring the macroblock.

[00123] It should be noted that although the motion vector assignment process in FIG. 6 begins with examining whether the F frame macroblock is an overlapped area macroblock, the process should be understood to apply equally where the F frame macroblock is a hole area macroblock or where there is only one motion vector passing through the F frame macroblock and such changes to the process flow are well within the capabilities of those of ordinary skill in the art to implement.

[00124] FIG. 8 illustrates a pixel-based, rather than a block-based, motion vector assignment process for FRUC system 300, where, similar to step 604 of FIG. 6, it is determined in step 806 whether the content maps exist for the reference frames. If so, then similar to step 606 of FIG. 6, any neighboring motion vectors starting from and ending in different content classes are eliminated in step 808. Operation then continues with step 816, where it is determined the pixel is overlapped. If so, then operation continues with step 818, where a median motion vector, based on a median of the overlapped motion vectors that have not been eliminated based on the different content class ending points, is assigned to the pixel. In another embodiment, as noted in step 820, an average motion vector, based on an average of the overlapped motion vectors that have not been eliminated based on the different content class ending points, is assigned to the pixel.

[00125] Where the F frame pixel is not an overlapped area, then in step 828 it is determined if a single motion vector passes through the pixel. If so, then operation continues with step 830, where the pixel is assigned with the motion vector passing

through it. Otherwise, the pixel is determined to be hole area and operation continues with step 822, where a median motion vector, based on a median of neighboring motion vectors is assigned to the pixel. In another embodiment, as noted in step 820, an average motion vector, based on an average of the neighboring motion vectors, is assigned to the pixel. It should be noted that the neighboring motion vectors that may be used in the median and average motion vector assignment may be subject to the elimination based on the different starting and ending point content classes.

[00126] FIG. 9 shows a block diagram of an access terminal 902x and an access point 904x in a wireless system. An “access terminal,” as discussed herein, refers to a device providing voice and/or data connectivity to a user. The access terminal may be connected to a computing device such as a laptop computer or desktop computer, or it may be a self contained device such as a personal digital assistant. The access terminal can also be referred to as a subscriber unit, mobile station, mobile, remote station, remote terminal, user terminal, user agent, or user equipment. The access terminal may be a subscriber station, wireless device, cellular telephone, PCS telephone, a cordless telephone, a Session Initiation Protocol (SIP) phone, a wireless local loop (WLL) station, a personal digital assistant (PDA), a handheld device having wireless connection capability, or other processing device connected to a wireless modem. An “access point,” as discussed herein, refers to a device in an access network that communicates over the air-interface, through one or more sectors, with the access terminals. The access point acts as a router between the access terminal and the rest of the access network, which may include an IP network, by converting received air-interface frames to IP packets. The access point also coordinates the management of attributes for the air interface.

[00127] For the reverse link, at access terminal 902x, a transmit (TX) data processor 914 receives traffic data from a data buffer 912, processes (e.g., encodes, interleaves, and symbol maps) each data packet based on a selected coding and modulation scheme, and provides data symbols. A data symbol is a modulation symbol for data, and a pilot symbol is a modulation symbol for pilot (which is known a priori). A modulator 916 receives the data symbols, pilot symbols, and possibly signaling for the reverse link, performs (e.g., OFDM) modulation and/or other processing as specified by the system, and provides a stream of output chips. A transmitter unit (TMTR) 918 processes (e.g.,

converts to analog, filters, amplifies, and frequency upconverts) the output chip stream and generates a modulated signal, which is transmitted from an antenna 920.

[00128] At access point 904x, the modulated signals transmitted by access terminal 902x and other terminals in communication with access point 904x are received by an antenna 952. A receiver unit (RCVR) 954 processes (e.g., conditions and digitizes) the received signal from antenna 952 and provides received samples. A demodulator (Demod) 956 processes (e.g., demodulates and detects) the received samples and provides detected data symbols, which are noisy estimate of the data symbols transmitted by the terminals to access point 904x. A receive (RX) data processor 958 processes (e.g., symbol demaps, deinterleaves, and decodes) the detected data symbols for each terminal and provides decoded data for that terminal.

[00129] For the forward link, at access point 904x, traffic data is processed by a TX data processor 960 to generate data symbols. A modulator 962 receives the data symbols, pilot symbols, and signaling for the forward link, performs (e.g., OFDM) modulation and/or other pertinent processing, and provides an output chip stream, which is further conditioned by a transmitter unit 964 and transmitted from antenna 952. The forward link signaling may include power control commands generated by a controller 970 for all terminals transmitting on the reverse link to access point 904x. At access terminal 902x, the modulated signal transmitted by access point 904x is received by antenna 920, conditioned and digitized by a receiver unit 922, and processed by a demodulator 924 to obtain detected data symbols. An RX data processor 926 processes the detected data symbols and provides decoded data for the terminal and the forward link signaling. Controller 930 receives the power control commands, and controls data transmission and transmit power on the reverse link to access point 904x. Controllers 930 and 970 direct the operation of access terminal 902x and access point 904x, respectively. Memory units 932 and 972 store program codes and data used by controllers 930 and 970, respectively.

[00130] The disclosed embodiments may be applied to any one or combinations of the following technologies: Code Division Multiple Access (CDMA) systems, Multiple-Carrier CDMA (MC-CDMA), Wideband CDMA (W-CDMA), High-Speed Downlink Packet Access (HSDPA), Time Division Multiple Access (TDMA) systems, Frequency Division Multiple Access (FDMA) systems, and Orthogonal Frequency Division Multiple Access (OFDMA) systems.

[00131] The steps of a method or algorithm described in connection with the embodiments disclosed herein may be embodied directly in hardware, in a software module executed by a processor, or in a combination of the two. A software module may reside in RAM memory, flash memory, ROM memory, EPROM memory, EEPROM memory, registers, a hard disk, a removable disk, a CD-ROM, or any other form of storage medium known in the art. An exemplary storage medium is coupled to the processor, such that the processor can read information from, and write information to, the storage medium. In the alternative, the storage medium may be integral to the processor. The processor and the storage medium may reside in an ASIC. The ASIC may reside in a user terminal. In the alternative, the processor and the storage medium may reside as discrete components in a user terminal.

[00132] It should be noted that the methods described herein may be implemented on a variety of communication hardware, processors and systems known by one of ordinary skill in the art. For example, the general requirement for the client to operate as described herein is that the client has a display to display content and information, a processor to control the operation of the client and a memory for storing data and programs related to the operation of the client. In one embodiment, the client is a cellular phone. In another embodiment, the client is a handheld computer having communications capabilities. In yet another embodiment, the client is a personal computer having communications capabilities. In addition, hardware such as a GPS receiver may be incorporated as necessary in the client to implement the various embodiments described herein. The various illustrative logics, logical blocks, modules, and circuits described in connection with the embodiments disclosed herein may be implemented or performed with 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 device, discrete gate or transistor logic, discrete hardware components, or any combination thereof designed to perform the functions described herein. A general-purpose processor may be a microprocessor, but, in the alternative, the processor may be any conventional processor, controller, microcontroller, or state machine. A processor may also be implemented as a combination of computing devices, e.g., a combination of a DSP and a microprocessor, a plurality of microprocessors, one or more microprocessors in conjunction with a DSP core, or any other such configuration.

[00133] The various illustrative logics, logical blocks, modules, and circuits described in connection with the embodiments disclosed herein may be implemented or performed with 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 device, discrete gate or transistor logic, discrete hardware components, or any combination thereof designed to perform the functions described herein. A general-purpose processor may be a microprocessor, but, in the alternative, the processor may be any conventional processor, controller, microcontroller, or state machine. A processor may also be implemented as a combination of computing devices, e.g., a combination of a DSP and a microprocessor, a plurality of microprocessors, one or more microprocessors in conjunction with a DSP core, or any other such configuration.

[00134] The embodiments described above are exemplary embodiments. Those skilled in the art may now make numerous uses of, and departures from, the above-described embodiments without departing from the inventive concepts disclosed herein. Various modifications to these embodiments may be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other embodiments, e.g., in an instant messaging service or any general wireless data communication applications, without departing from the spirit or scope of the novel aspects described herein. Thus, the scope of the novel aspects of the embodiments is not intended to be limited to the embodiments shown herein but is to be accorded the widest scope consistent with the principles and novel features disclosed herein. The word “exemplary” is used exclusively herein to mean “serving as an example, instance, or illustration.” Any embodiment described herein as “exemplary” is not necessarily to be construed as preferred or advantageous over other embodiments. Accordingly, the scope of the novel aspects of the embodiments is to be defined solely by the scope of the following claims.

CLAIMS

What is claimed is:

Method

1. A method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, the method comprising:
 - partitioning the video frame to be interpreted into a plurality of areas;
 - determining a number of motion vectors passing through one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame; and
 - generating a motion vector for assigning to the one area based on the number of motion vectors passing through the one area.
2. The method of claim 1, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of non-overlapping areas.
3. The method of claim 1, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of overlapping areas.
4. The method of claim 1, wherein generating the motion vector to the one area based on the number of motion vectors passing through the one area comprises selecting one of the number of motion vectors passing through the one area.
5. The method of claim 1, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of a fixed size.

6. The method of claim 1, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of different sizes.

7. The method of claim 1, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of pixels.

8. The method of claim 1, wherein determining the number of motion vectors passing through the one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame comprises:

for each motion vector passing through the one area:

determining a motion vector starting point content class and a motion vector ending point content class;

comparing the motion vector starting point content class and the motion vector ending point content class; and

removing the motion vector from being considered in the number of motion vectors passing through the one area if the motion vector starting point content class and the motion vector ending point content class are different.

9. The method of claim 1, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a zero motion vector, (ii) an average of any available neighboring motion vectors, and (iii) a median of any available neighboring motion vectors; if the number of motion vectors passing through the one area is zero.

10. The method of claim 1, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a center pixel motion vector, (ii) a most covered area motion vector, (iii) a weighted motion vector, and (iv) a median motion vector; if the number of motion vectors passing through the one area is greater than one.

11. The method of claim 1, wherein the content map comprises a plurality of content classes.
12. The method of claim 11, wherein the plurality of content classes comprises a background content class.
13. The method of claim 12, wherein the background class comprises an appearing object class, a disappearing object class and a static background class.
14. The method of claim 11, wherein the plurality of content classes comprises a moving object content class.
15. The method of claim 11, wherein the plurality of content classes comprises an edge content class.
16. A computer readable medium having instructions stored thereon for causing a computer to execute a method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, the method comprising:
 - partitioning the video frame to be interpreted into a plurality of areas;
 - determining a number of motion vectors passing through one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame; and
 - generating a motion vector for assigning to the one area based on the number of motion vectors passing through the one area.
17. The computer readable medium of claim 16, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of non-overlapping areas.

18. The computer readable medium of claim 16, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of overlapping areas.

19. The computer readable medium of claim 16, wherein generating the motion vector to the one area based on the number of motion vectors passing through the one area comprises selecting one of the number of motion vectors passing through the one area.

20. The computer readable medium of claim 16, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of a fixed size.

21. The computer readable medium of claim 16, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of different sizes.

22. The computer readable medium of claim 16, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of pixels.

23. The computer readable medium of claim 16, wherein determining the number of motion vectors passing through the one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame comprises:

for each motion vector passing through the one area:

determining a motion vector starting point content class and a motion vector ending point content class;

comparing the motion vector starting point content class and the motion vector ending point content class; and

removing the motion vector from being considered in the number of motion vectors passing through the one area if the motion vector starting point content class and the motion vector ending point content class are different.

24. The computer readable medium of claim 16, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a zero motion vector, (ii) an average of any available neighboring motion vectors, and (iii) a median of any available neighboring motion vectors; if the number of motion vectors passing through the one area is zero.

25. The computer readable medium of claim 16, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a center pixel motion vector, (ii) a most covered area motion vector, (iii) a weighted motion vector, and (iv) a median motion vector; if the number of motion vectors passing through the one area is greater than one.

26. The computer readable medium of claim 16, wherein the content map comprises a plurality of content classes.

27. The computer readable medium of claim 26 wherein the plurality of content classes comprises a background content class.

28. The computer readable medium of claim 27, wherein the background class comprises an appearing object class, a disappearing object class and a static background class.

29. The computer readable medium of claim 26, wherein the plurality of content classes comprises a moving object content class.

30. The computer readable medium of claim 26, wherein the plurality of content classes comprises an edge content class.

31. An apparatus for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, comprising:

means for partitioning the video frame to be interpreted into a plurality of areas;

means for determining a number of motion vectors passing through one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame; and

means for generating a motion vector for assigning to the one area based on the number of motion vectors passing through the one area.

32. The apparatus of claim 31, wherein the means for partitioning the video frame to be interpreted into the plurality of areas comprises means for partitioning the video frame to be interpreted into a plurality of non-overlapping areas.

33. The apparatus of claim 31, wherein the means for partitioning the video frame to be interpreted into the plurality of areas comprises means for partitioning the video frame to be interpreted into a plurality of overlapping areas.

34. The apparatus of claim 31, wherein the means for generating the motion vector to the one area based on the number of motion vectors passing through the one area comprises means for selecting one of the number of motion vectors passing through the one area.

35. The apparatus of claim 31, wherein the means for partitioning the video frame to be interpreted into the plurality of areas comprises means for partitioning the video frame to be interpreted into a plurality of areas of a fixed size.

36. The apparatus of claim 31, wherein the means for partitioning the video frame to be interpreted into the plurality of areas comprises means for partitioning the video frame to be interpreted into a plurality of areas of different sizes.

37. The apparatus of claim 31, wherein the means for partitioning the video frame to be interpreted into the plurality of areas comprises means for partitioning the video frame to be interpreted into a plurality of pixels.

38. The apparatus of claim 31 wherein the means for determining the number of motion vectors passing through the one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame comprises:

means for determining a motion vector starting point content class and a motion vector ending point content class;

means for comparing the motion vector starting point content class and the motion vector ending point content class; and

means for removing the motion vector from being considered in the number of motion vectors passing through the one area if the motion vector starting point content class and the motion vector ending point content class are different.

39. The apparatus of claim 31, wherein the means for generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises means for assigning the one area with a motion vector selected from a group consisting of: (i) a zero motion vector, (ii) an average of any available neighboring motion vectors, and (iii) a median of any available neighboring motion vectors; if the number of motion vectors passing through the one area is zero.

40. The apparatus of claim 31, wherein the means for generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises means for assigning the one area with a motion vector selected from a group consisting of: (i) a center pixel motion vector, (ii) a most covered area motion vector, (iii) a weighted motion vector, and (iv) a median motion vector; if the number of motion vectors passing through the one area is greater than one.

41. The apparatus of claim 31, wherein the content map comprises a plurality of content classes.

42. The apparatus of claim 41, wherein the plurality of content classes comprises a background content class.

43. The apparatus of claim 42, wherein the background class comprises an appearing object class, a disappearing object class and a static background class.

44. The apparatus of claim 41, wherein the plurality of content classes comprises a moving object content class.

45. The apparatus of claim 41, wherein the plurality of content classes comprises an edge content class.

46. At least one processor configured to implement a method for processing a plurality of motion vectors for a video frame to be interpolated using a first reference frame and a second reference frame, each reference frame having a content map, the method comprising:

partitioning the video frame to be interpreted into a plurality of areas;

determining a number of motion vectors passing through one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame; and

generating a motion vector for assigning to the one area based on the number of motion vectors passing through the one area.

47. The at least one processor of claim 46, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of non-overlapping areas.

48. The at least one processor of claim 46, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of overlapping areas.

49. The at least one processor of claim 46, wherein generating the motion vector to the one area based on the number of motion vectors passing through the one area comprises selecting one of the number of motion vectors passing through the one area.

50. The at least one processor of claim 46, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of a fixed size.

51. The at least one processor of claim 46, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of areas of different sizes.

52. The at least one processor of claim 46, wherein partitioning the video frame to be interpreted into the plurality of areas comprises partitioning the video frame to be interpreted into a plurality of pixels.

53. The at least one processor of claim 46, wherein determining the number of motion vectors passing through the one area of the plurality of areas based on the content maps of the first reference frame and the second reference frame comprises:

for each motion vector passing through the one area:

determining a motion vector starting point content class and a motion vector ending point content class;

comparing the motion vector starting point content class and the motion vector ending point content class; and

removing the motion vector from being considered in the number of motion vectors passing through the one area if the motion vector starting point content class and the motion vector ending point content class are different.

54. The at least one processor of claim 46, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a zero motion vector, (ii) an average of any available neighboring

motion vectors, and (iii) a median of any available neighboring motion vectors; if the number of motion vectors passing through the one area is zero.

55. The at least one processor of claim 46, wherein generating the motion vector for assigning to the one area based on the number of motion vectors passing through the one area comprises assigning the one area with a motion vector selected from a group consisting of: (i) a center pixel motion vector, (ii) a most covered area motion vector, (iii) a weighted motion vector, and (iv) a median motion vector; if the number of motion vectors passing through the one area is greater than one.

56. The at least one processor of claim 46, wherein the content map comprises a plurality of content classes.

57. The at least one processor of claim 56, wherein the plurality of content classes comprises a background content class.

58. The at least one processor of claim 57, wherein the background class comprises an appearing object class, a disappearing object class and a static background class.

59. The at least one processor of claim 56, wherein the plurality of content classes comprises a moving object content class.

60. The at least one processor of claim 56, wherein the plurality of content classes comprises an edge content class.