

US Patent & Trademark Office

Patent Public Search | Text View

United States Patent	12395665
Kind Code	B2
Date of Patent	August 19, 2025
Inventor(s)	Furht; Borivoje et al.

Encoder and bitstream for efficient coding of global motion vectors

Abstract

A video encoder is configured to generate a bitstream for decoding by a compatible decoder which includes circuitry configured to receive a bitstream, extract a residual of a control point motion vector for a current frame and from the bitstream, and combine the residual of the control point motion vector with a prediction of the control point motion vector for the current frame.

Inventors: Furht; Borivoje (Boca Raton, FL), Kalva; Hari (Boca Raton, FL), Adzic; Velibor (Canton, GA)

Applicant: OP Solutions, LLC (Amherst, MA)

Family ID: 1000008763459

Appl. No.: 18/379313

Filed: October 12, 2023

Prior Publication Data

Document Identifier	Publication Date
US 20240048750 A1	Feb. 08, 2024

Related U.S. Application Data

continuation parent-doc US 17874918 20220727 US 11800137 child-doc US 18379313
continuation parent-doc US 17006613 20200828 US 11438620 20220906 child-doc US 17874918
continuation parent-doc WO PCT/US2020/029936 20200424 PENDING child-doc US 17006613
us-provisional-application US 62838521 20190425

Publication Classification

Int. Cl.: H04N19/51 (20140101); H04N19/124 (20140101); H04N19/172 (20140101);
H04N19/593 (20140101); H04N19/80 (20140101)

U.S. Cl.:

CPC H04N19/51 (20141101); H04N19/124 (20141101); H04N19/172 (20141101);
H04N19/593 (20141101); H04N19/80 (20141101);

Field of Classification Search

CPC: H04N (19/51)

References Cited

U.S. PATENT DOCUMENTS

Patent No.	Issued Date	Patentee Name	U.S. Cl.	CPC
11082713	12/2020	Liu et al.	N/A	N/A
2005/0185937	12/2004	Comer	386/334	H04N 21/234327
2018/0192047	12/2017	Lv	N/A	H04N 19/527
2018/0192069	12/2017	Chen	N/A	H04N 19/176
2020/0221120	12/2019	Robert	N/A	H04N 19/521
2021/0160533	12/2020	Zhang	N/A	H04N 19/51
2023/0081842	12/2022	Wang	375/240.02	H04N 19/70

OTHER PUBLICATIONS

Invitation to respond to written opinion in corresponding Singapore application 11202111762W dated Apr. 5, 2024. cited by applicant
JVET-K0465-v2, 11th Meeting Ljubljana, SI, Jul. 10-18, 2018, JVET of ITU-T SG 16 WP3 and ISO/IEC JTC 1/SC29/WG11, “Draft text for affine motion compensation”. cited by applicant
JVET-N1001-v1, “Versatile Video Coding (Draft 5)”, 14th Meeting: Geneva, CH, Mar. 19-27, 2019. cited by applicant
Notice of Reasons for Rejection (English translation) in corresponding Japanese patent application 2021=563020, dated Dec. 2, 2022. cited by applicant
First Office Action (English translation) in corresponding application in China, app. No. 2020800458865. cited by applicant

Primary Examiner: Jean Baptiste; Jerry T

Attorney, Agent or Firm: OP Solutions IP Department

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS (1) This application is a continuation of co-pending U.S. patent application Ser. No. 17/874,918, filed on Jul. 27, 2022, and entitled EFFICIENT CODING OF GLOBAL MOTION VECTORS, which is a continuation of U.S. patent application Ser. No. 17,006,613, filed on Aug. 8, 2020 and entitled EFFICIENT CODING OF GLOBAL MOTION VECTORS, now issued as U.S. Pat. No. 11,438,620, which claims the benefit of priority of International Application No. PCT/US20/29936, filed on Apr. 24, 2020 and entitled

“EFFICIENT CODING OF GLOBAL MOTION VECTORS,” which claims the benefit of priority of U.S. Provisional Patent Application Ser. No. 62/838,521, filed on Apr. 25, 2019, and titled “EFFICIENT CODING OF GLOBAL MOTION VECTORS.” Each of U.S. patent application Ser. No. 17/006,613, International Application No. PCT/US20/29936 and U.S. Provisional Patent Application Ser. No. 62/838,521 is incorporated by reference herein in its entirety.

FIELD OF THE INVENTION

(1) The present invention generally relates to the field of video compression. In particular, the present invention is directed to efficient coding of global motion vectors.

BACKGROUND

(2) A video codec can include an electronic circuit or software that compresses or decompresses digital video. It can convert uncompressed video to a compressed format or vice versa. In the context of video compression, a device that compresses video (and/or performs some function thereof) can typically be called an encoder, and a device that decompresses video (and/or performs some function thereof) can be called a decoder.

(3) A format of the compressed data can conform to a standard video compression specification. The compression can be lossy in that the compressed video lacks some information present in the original video. A consequence of this can include that decompressed video can have lower quality than the original uncompressed video because there is insufficient information to accurately reconstruct the original video.

(4) There can be complex relationships between the video quality, the amount of data used to represent the video (e.g., determined by the bit rate), the complexity of the encoding and decoding algorithms, sensitivity to data losses and errors, ease of editing, random access, end-to-end delay (e.g., latency), and the like.

(5) Motion compensation can include an approach to predict a video frame or a portion thereof given a reference frame, such as previous and/or future frames, by accounting for motion of the camera and/or objects in the video. It can be employed in the encoding and decoding of video data for video compression, for example in the encoding and decoding using the Motion Picture Experts Group (MPEG)-2 (also referred to as advanced video coding (AVC) and H.264) standard. Motion compensation can describe a picture in terms of the transformation of a reference picture to the current picture. The reference picture can be previous in time when compared to the current picture, from the future when compared to the current picture. When images can be accurately synthesized from previously transmitted and/or stored images, compression efficiency can be improved.

SUMMARY OF THE DISCLOSURE

(6) In an aspect, a decoder includes circuitry configured to receive a bitstream, extract a residual of a control point motion vector for a current frame from the bitstream, and combine the residual of the control point motion vector with a prediction of the control point motion vector for the current frame.

(7) In another aspect, a method includes receiving, by a decoder, a bitstream. The method includes extracting a residual of a control point motion vector for a current frame and from the bitstream. The method includes combining the residual of the control point motion vector with a prediction of the control point motion vector for the current frame.

(8) These and other aspects and features of non-limiting embodiments of the present invention will become apparent to those skilled in the art upon review of the following description of specific non-limiting embodiments of the invention in conjunction with the accompanying drawings.

Description

BRIEF DESCRIPTION OF THE DRAWINGS

- (1) For the purpose of illustrating the invention, the drawings show aspects of one or more embodiments of the invention. However, it should be understood that the present invention is not limited to the precise arrangements and instrumentalities shown in the drawings, wherein:
- (2) FIG. 1 is a diagram illustrating motion vectors of an example frame with global and local motion;
- (3) FIG. 2 illustrates three example motion models that can be utilized for global motion including their index value (0, 1, or 2);
- (4) FIG. 3 is a process flow diagram according to some example implementations of the current subject matter;
- (5) FIG. 4 is a system block diagram of an example decoder according to some example implementations of the current subject matter;
- (6) FIG. 5 is a process flow diagram according to some example implementations of the current subject matter;
- (7) FIG. 6 is a system block diagram of an example encoder according to some example implementations of the current subject matter; and
- (8) FIG. 7 is a block diagram of a computing system that can be used to implement any one or more of the methodologies disclosed herein and any one or more portions thereof.
- (9) The drawings are not necessarily to scale and may be illustrated by phantom lines, diagrammatic representations and fragmentary views. In certain instances, details that are not necessary for an understanding of the embodiments or that render other details difficult to perceive may have been omitted.

DETAILED DESCRIPTION

(10) “Global motion” in video refers to motion and/or a motion model common to all pixels of a region, where a region may be a picture, a frame, or any portion of a picture or frame such as a block, CTU, or other subset of contiguous pixels. Global motion may be caused by camera motion; for example and without limitation, camera panning and zooming may create motion in a frame that may typically affect the entire frame. Motion present in portions of a video may be referred to as local motion. Local motion may be caused by moving objects in a scene, such as without limitation an object moving from left to right in the scene. Videos may contain a combination of local and global motion. Some implementations of the current subject matter may provide for efficient approaches to communicate global motion to the decoder and use of global motion vectors in improving compression efficiency.

(11) FIG. 1 is a diagram illustrating exemplary embodiments of motion vectors of an exemplary frame **100** with global and local motion. Frame **100** may include a number of blocks of pixels illustrated as squares, and their associated motion vectors illustrated as arrows. Squares (e.g., blocks of pixels) with arrows pointing up and to the left may indicate blocks with motion that may be considered to be global motion and squares with arrows pointing in other directions (indicated by **104**) indicate blocks with local motion. In the illustrated example of FIG. 1, many of the blocks have same global motion. Signaling global motion in a header, such as a picture parameter set (PPS) or sequence parameter set (SPS), and using signaled global motion may reduce motion vector information needed by blocks and may result in improved prediction. Although for illustrative purposes examples described below refer to determination and/or application of global or local motion vectors at a block level, global motion vectors may be determined and/or applied for any region of a frame and/or picture, including regions made up of multiple blocks, regions bounded by any geometric form such as without limitation regions defined by geometric and/or exponential coding in which one or more lines and/or curves bounding the shape may be angled and/or curved, and/or an entirety of a frame and/or picture. Although signaling is described herein as being performed at a frame level and/or in a header and/or parameter set of a frame, signaling

may alternatively or additionally be performed at a sub-picture level, where a sub-picture may include any region of a frame and/or picture as described above.

(12) As an example, and still referring to FIG. 1, simple translational motion may be described using a motion vector (MV) with two components MV.sub.x, MV.sub.y that describes displacement of blocks and/or pixels in a current frame. More complex motion such as rotation, zooming, and warping may be described using affine motion vectors, where an “affine motion vector,” as used in this disclosure, is a vector describing a uniform displacement of a set of pixels or points represented in a video picture and/or picture, such as a set of pixels illustrating an object moving across a view in a video without changing apparent shape during motion. Some approaches to video encoding and/or decoding may use 4-parameter or 6-parameter affine models for motion compensation in inter picture coding.

(13) For example, a six parameter affine motion may be described as:

$$x'=ax+by+c$$

$$y'=dx+ey+f$$

(14) A four parameter affine motion may be described as:

$$x'=ax+by+c$$

$y'=-bx+ay+f$ where (x,y) and (x',y') are pixel locations in current and reference pictures, respectively; a, b, c, d, e, and f are the parameters of the affine motion model.

(15) With continued reference to FIG. 1, parameters used describe affine motion may be signaled to a decoder to apply affine motion compensation at the decoder. In some approaches, motion parameters may be signaled explicitly or by signaling translational control point motion vectors (CPMVs) and then deriving the affine motion parameters from translational motion vectors. Two control point motion vectors (CPMVs) may be utilized to derive affine motion parameters for a four-parameter affine motion model and three control point translational motion vectors (CPMVs) may be utilized to obtain parameters for a six-parameter motion model. Signaling affine motion parameters using control point motion vectors may allow use of efficient motion vector coding methods to signal affine motion parameters.

(16) In some implementations, and still referring to FIG. 1, global motion signaling may be included in a header, such as the PPS or SPS. Global motion may vary from picture to picture. Motion vectors signaled in picture headers may describe motion relative to previously decoded frames. In some implementations, global motion may be translational or affine. Motion model (e.g., number of parameters, whether the model is affine, translational, or other) used may also be signaled in a picture header. FIG. 2 illustrates three example motion models 200 that may be utilized for global motion including their index value (0, 1, or 2).

(17) Continuing to refer to FIG. 2, PPSs may be used to signal parameters that may change between pictures of a sequence. Parameters that remain the same for a sequence of pictures may be signaled in a sequence parameter set to reduce the size of PPS and reduce video bitrate. An example picture parameter set (PPS) is shown in table 1:

(18) TABLE-US-00001 Descriptor pic_parameter_set_rbsp() { pps_pic_parameter_set_id ue(v)
pps_seq_parameter_set_id u(4) mixed_nalu_types_in_pic_flag u(1)
pic_width_in_luma_samples ue(v) pic_height_in_luma_samples ue(v)
pps_conformance_window_flag u(1) if(pps_conformance_window_flag) {
pps_conf_win_left_offset ue(v) pps_conf_win_right_offset ue(v)
pps_conf_win_top_offset ue(v) pps_conf_win_bottom_offset ue(v) }
scaling_window_explicit_signalling_flag u(1) if(scaling_window_explicit_signalling_flag) {
scaling_win_left_offset ue(v) scaling_win_right_offset ue(v)
scaling_win_top_offset ue(v) scaling_win_bottom_offset ue(v) }
output_flag_present_flag u(1) subpic_id_mapping_in_pps_flag u(1) if(
subpic_id_mapping_in_pps_flag) { pps_num_subpics_minus1 ue(v)
pps_subpic_id_len_minus1 ue(v) for(i = 0; i <= pps_num_subpic_minus1; i++)

```

pps_subpic_id[ i ] u(v) } no_pic_partition_flag u(1) if( !no_pic_partition_flag ) {
pps_log2_ctu_size_minus5 u(2) num_exp_tile_columns_minus1 ue(v)
num_exp_tile_rows_minus1 ue(v) for( i = 0; i <= num_exp_tile_columns_minus1; i++ )
    tile_column_width_minus1 [ i ] ue(v) for( i = 0; i <= num_exp_tile_rows_minus1; i++
)    tile_row_height_minus1 [ i ] ue(v) if( NumTilesInPic > 1) rect_slice_flag
u(1) if( rect_slice_flag ) single_slice_per_subpic_flag u(1) if( rect_slice_flag &&
!single_slice_per_subpic_flag ) { num_slices_in_pic_minus1 ue(v) if(
num_slices_in_pic_minus1 > 0 ) tile_idx_delta_present_flag u(1) for( i = 0; i <
num_slices_in_pic_minus1; i++ ) { if( NumTileColumns > 1 )
slice_width_in_tiles_minus1 [ i ] ue(v) if( NumTileRows > 1 && (
tile_idx_delta_present_flag || SliceTopLeftTileIdx[ i ] % NumTileColumns == 0 ) )
    slice_height_in_tiles_minus1[ i ] ue(v) if( slice_width_in_tiles_minus1[ i ] ==
0 && slice_height_in_tiles_minus1[ i ] == 0 && RowHeight[
SliceTopLeftTileIdx[ i ] / NumTileColumns ] > 1 ) {
num_exp_slices_in_tile[ i ] ue(v) for( j = 0; j < num_exp_slices_in_tile[ i ]; j++ )
    exp_slice_height_in_ctus_minus1 [ i ][ j ] ue(v) i += NumSlicesInTile[ i
] - 1 } if( tile_idx_delta_present_flag && i < num_slices_in_pic_minus1 )
    tile_idx_delta[ i ] se(v) } } loop_filter_across_tiles_enabled_flag u(1)
loop_filter_across_slices_enabled_flag u(1) } cabac_init_present_flag u(1) for( i = 0;
i < 2; i++ ) num_ref_idx_default_active_minus1 [ i ] ue(v) rpl1_idx_present_flag u(1)
init_qp_minus26 se(v) cu_qp_delta_enabled_flag u(1) pps_chroma_tool_offsets_present_flag
u(1) if( pps_chroma_tool_offsets_present_flag ) { pps_cb_qp_offset se(v)
pps_cr_qp_offset se(v) pps_joint_cbc_r_qp_offset_present_flag u(1) if(
pps_joint_cbc_r_qp_offset_present_flag ) pps_joint_cbc_r_qp_offset_value se(v)
pps_slice_chroma_qp_offsets_present_flag u(1) pps_cu_chroma_qp_offset_list_enabled_flag
u(1) } if( pps_cu_chroma_qp_offset_list_enabled_flag ) {
chroma_qp_offset_list_len_minus1 ue(v) for( i = 0; i <= chroma_qp_offset_list_len_minus1;
i++ ) { cb_qp_offset_list[ i ] se(v) cr_qp_offset_list[ i ] se(v) if(
pps_joint_cbc_r_qp_offset_present_flag ) joint_cbc_r_qp_offset_list[ i ] se(v) } }
pps_weighted_pred_flag u(1) pps_weighted_bipred_flag u(1)
deblocking_filter_control_present_flag u(1) if( deblocking_filter_control_present_flag ) {
deblocking_filter_override_enabled_flag u(1) pps_deblocking_filter_disabled_flag u(1)
if( !pps_deblocking_filter_disabled_flag ) { pps_beta_offset_div2 se(v)
pps_tc_offset_div2 se(v) pps_cb_beta_offset_div2 se(v) pps_cb_tc_offset_div2
se(v) pps_cr_beta_offset_div2 se(v) pps_cr_tc_offset_div2 se(v) } }
rpl_info_in_ph_flag u(1) if( deblocking_filter_override_enabled_flag ) dbf_info_in_ph_flag
u(1) sao_info_in_ph_flag u(1) alf_info_in_ph_flag u(1) if( ( pps_weighted_pred_flag ||
pps_weighted_bipred_flag ) && rpl_info_in_ph_flag ) wp_info_in_ph_flag u(1)
qp_delta_info_in_ph_flag u(1) pps_ref_wraparound_enabled_flag u(1) if(
pps_ref_wraparound_enabled_flag ) pps_ref_wraparound_offset ue(v)
picture_header_extension_present_flag u(1) slice_header_extension_present_flag u(1)
pps_extension_flag u(1) if( pps_extension_flag ) while( more_rbsp_data( ) )
pps_extension_data_flag u(1) rbsp_trailing_bits( ) }

```

(19) Still referring to FIG. 2, additional fields may be added to the PPS to signal global motion. In case of global motion, the presence of global motion parameters in a sequence of pictures can be signaled in a SPS and the PPS references the SPS by SPS ID. The SPS in some approaches to decoding may be modified to add a field to signal presence of global motion parameters in SPS. For example a one-bit field may be added to the SPS. If global_motion_present bit is 1, global motion related parameters may be expected in a PPS; if global_motion_present bit is 0, no global motion parameter related fields may be present in the PPS. For example, the PPS of table 1 may be

extended to include a global_motion_present field, for example, as shown in table 2:

(20) TABLE-US-00002 Descriptor sequence_parameter_set_rbsp() {

sps_sequence_parameter_set_id ue(v) . . . global_motion_present u(1) rbsp_trailing_bits() }

(21) Similarly, the PPS may include a pps_global_motion_parameters field for a frame, for example as shown in table 3:

(22) TABLE-US-00003 Descriptor pic_parameter_set_rbsp() { pps_pic_parameter_set_id ue(v)

pps_seq_parameter_set_id ue(v) . . . pps_global_motion_parameters) rbsp_trailing_bits() }

(23) In more detail, the PPS may include fields to characterize global motion parameters using control point motion vectors, for example as shown in table 4:

(24) TABLE-US-00004 Descriptor pps_global_motion_parameters () { motion_model_used u(2)

mv0_x se(v) mv1_y se(v) if(motion_model_used == 1){ mv1_x se(v) mv1_y se(v) }

if(motion_model_used == 2){ mv2_x se(v) mv2_y se(v) } }

(25) As a further non-limiting example, Table 5 below may represent an exemplary SPS:

(26) TABLE-US-00005 Descriptor seq_parameter_set_rbsp() { sps_seq_parameter_set_id u(4)

sps_video_parameter_set_id u(4) sps_max_sublayers_minus1 u(3) sps_reserved_zero_4bits

u(4) sps_ptl_dpb_hrd_params_present_flag u(1) if(sps_ptl_dpb_hrd_params_present_flag)

profile_tier_level(1, sps_max_sublayers_minus1) gdr_enabled_flag u(1)

chroma_format_idc u(2) if(chroma_format_idc == 3) separate_colour_plane_flag u(1)

res_change_in_clvs_allowed_flag u(1) pic_width_max_in_luma_samples ue(v)

pic_height_max_in_luma_samples ue(v) sps_conformance_window_flag u(1) if(

sps_conformance_window_flag) { sps_conf_win_left_offset ue(v)

sps_conf_win_right_offset ue(v) sps_conf_win_top_offset ue(v)

sps_conf_win_bottom_offset ue(v) } sps_log2_ctu_size_minus5 u(2)

subpic_info_present_flag u(1) if(subpic_info_present_flag) { sps_num_subpics_minus1

ue(v) sps_independent_subpics_flag u(1) for(i = 0; sps_num_subpics_minus1 > 0 && i

<= sps_num_subpics_minus1; i++) { if(i > 0 && pic_width_max_in_luma_samples >

CtbSizeY) subpic_ctu_top_left_x[i] u(v) if(i > 0 &&

pic_height_max_in_luma_samples > CtbSizeY) { subpic_ctu_top_left_y[i] u(v) if(i

< sps_num_subpics_minus1 && pic_width_max_in_luma_samples > CtbSizeY)

subpic_width_minus1 [i] u(v) if(i < sps_num_subpics_minus1 &&

pic_height_max_in_luma_samples > CtbSizeY) subpic_height_minus1 [i] u(v)

if(!sps_independent_subpics_flag) { subpic_treated_as_pic_flag[i] u(1)

loop_filter_across_subpic_enabled_flag[i] u(1) } } sps_subpic_id_len_minus1

ue(v) subpic_id_mapping_explicitly_signalled_flag u(1) if(

subpic_id_mapping_explicitly_signalled_flag) { subpic_id_mapping_in_sps_flag u(1)

if(subpic_id_mapping_in_sps_flag) for(i = 0; i <= sps_num_subpics_minus1;

i++) sps_subpic_id[i] u(v) } } bit_depth_minus8 ue(v)

sps_entropy_coding_sync_enabled_flag u(1) if(sps_entropy_coding_sync_enabled_flag)

sps_wpp_entry_point_offsets_present_flag u(1) sps_weighted_pred_flag u(1)

sps_weighted_bipred_flag u(1) log2_max_pic_order_cnt_lsb_minus4 u(4) sps_poc_msb_flag

u(1) if(sps_poc_msb_flag) poc_msb_len_minus1 ue(v) num_extra_ph_bits_bytes u(2)

extra_ph_bits_struct(num_extra_ph_bits_bytes) num_extra_sh_bits_bytes u(2)

extra_sh_bits_struct(num_extra_sh_bits_bytes) if(sps_max_sublayers_minus1 > 0)

sps_sublayer_dpb_params_flag u(1) if(sps_ptl_dpb_hrd_params_present_flag)

dpb_parameters(sps_max_sublayers_minus1, sps_sublayer_dpb_params_flag)

long_term_ref_pics_flag u(1) inter_layer_ref_pics_present_flag u(1) sps_idr_rpl_present_flag

u(1) rpl1_same_as_rpl0_flag u(1) for(i = 0; i < rpl1_same_as_rpl0_flag ? 1 : 2; i++) {

num_ref_pic_lists_in_sps[i] ue(v) for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)

ref_pic_list_struct(i, j) } if(ChromaArrayType != 0) qtbtt_dual_tree_intra_flag u(1)

log2_min_luma_coding_block_size_minus2 ue(v) partition_constraints_override_enabled_flag

```

u(1)    sps_log2_diff_min_qt_min_cb_intra_slice_luma ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma ue(v)    if(
sps_max_mtt_hierarchy_depth_intra_slice_luma != 0 ) {
sps_log2_diff_max_bt_min_qt_intra_slice_luma ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma ue(v)    }
sps_log2_diff_min_qt_min_cb_inter_slice ue(v)    sps_max_mtt_hierarchy_depth_inter_slice
ue(v)    if( sps_max_mtt_hierarchy_depth_inter_slice != 0 ) {
sps_log2_diff_max_bt_min_qt_inter_slice ue(v)    sps_log2_diff_max_tt_min_qt_inter_slice
ue(v)    }    if( qtbt_dual_tree_intra_flag ) {
sps_log2_diff_min_qt_min_cb_intra_slice_chroma ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma ue(v)    if(
sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0 ) {
sps_log2_diff_max_bt_min_qt_intra_slice_chroma ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma ue(v)    }    }
sps_max_luma_transform_size_64_flag u(1)    if( ChromaArrayType != 0 ) {
sps_joint_cbr_enabled_flag u(1)    same_qp_table_for_chroma u(1)    numQpTables =
same_qp_table_for_chroma ? 1 : ( sps_joint_cbr_enabled_flag ? 3 : 2)    for( i = 0; i <
numQpTables; i++ ) {        qp_table_start_minus26[ i ] se(v)
num_points_in_qp_table_minus1[ i ] ue(v)        for( j = 0; j <=
num_points_in_qp_table_minus1[ i ]; j++ ) {            delta_qp_in_val_minus1 [ i ][ j ] ue(v)
            delta_qp_diff_val[ i ][ j ] ue(v)        }    }    }    sps_sao_enabled_flag u(1)
sps_alf_enabled_flag u(1)    if( sps_alf_enabled_flag && ChromaArrayType != 0 )
sps_ccalf_enabled_flag u(1)    sps_transform_skip_enabled_flag u(1)    if(
sps_transform_skip_enabled_flag ) {        log2_transform_skip_max_size_minus2 ue(v)
sps_bdpcm_enabled_flag u(1)    }    sps_ref_wraparound_enabled_flag u(1)
sps_temporal_mvp_enabled_flag u(1)    if( sps_temporal_mvp_enabled_flag )
sps_sbtmvp_enabled_flag u(1)    sps_amvr_enabled_flag u(1)    sps_bdof_enabled_flag u(1)    if(
sps_bdof_enabled_flag )        sps_bdof_pic_present_flag u(1)    sps_smvd_enabled_flag u(1)
sps_dmvr_enabled_flag u(1)    if( sps_dmvr_enabled_flag )        sps_dmvr_pic_present_flag u(1)
    sps_mmvd_enabled_flag u(1)    sps_isp_enabled_flag u(1)    sps_mrl_enabled_flag u(1)
sps_mip_enabled_flag u(1)    if( ChromaArrayType != 0 )        sps_cclm_enabled_flag u(1)    if(
chroma_format_idc == 1 ) {        sps_chroma_horizontal_collocated_flag u(1)
sps_chroma_vertical_collocated_flag u(1)    }    sps_mts_enabled_flag u(1)    if(
sps_mts_enabled_flag ) {        sps_explicit_mts_intra_enabled_flag u(1)
sps_explicit_mts_inter_enabled_flag u(1)    }    six_minus_max_num_merge_cand ue(v)
sps_sbt_enabled_flag u(1)    sps_affine_enabled_flag u(1)    if( sps_affine_enabled_flag ) {
five_minus_max_num_subblock_merge_cand ue(v)        sps_affine_type_flag u(1)        if(
sps_amvr_enabled_flag )            sps_affine_amvr_enabled_flag u(1)
sps_affine_prof_enabled_flag u(1)        if( sps_affine_prof_enabled_flag )
sps_prof_pic_present_flag u(1)    }    sps_palette_enabled_flag u(1)    if( ChromaArrayType == 3
&& !sps_max_luma_transform_size_64_flag )        sps_act_enabled_flag u(1)    if(
sps_transform_skip_enabled_flag || sps_palette_enabled_flag )        min_qp_prime_ts_minus4
ue(v)    sps_bcw_enabled_flag u(1)    sps_ibc_enabled_flag u(1)    if( sps_ibc_enabled_flag )
        six_minus_max_num_ibc_merge_cand ue(v)    sps_ciip_enabled_flag u(1)    if(
sps_mmvd_enabled_flag )        sps_fpel_mmvd_enabled_flag u(1)    if( MaxNumMergeCand >= 2
) {        sps_gpm_enabled_flag u(1)        if( sps_gpm_enabled_flag && MaxNumMergeCand >=
3)            max_num_merge_cand_minus_max_num_gpm_cand ue(v)        }
sps_lmcs_enabled_flag u(1)    sps_lfnst_enabled_flag u(1)    sps_ladf_enabled_flag u(1)    if(
sps_ladf_enabled_flag ) {        sps_num_ladf_intervals_minus2 u(2)
sps_ladf_lowest_interval_qp_offset se(v)        for( i = 0; i < sps_num_ladf_intervals_minus2 + 1;

```



```

i++ ) {      sps_ladf_qp_offset[ i ] se(v)      sps_ladf_delta_threshold_minus1 [ i ] ue(v)
    } } log2_parallel_merge_level_minus2 ue(v)      sps_explicit_scaling_list_enabled_flag
u(1)      sps_dep_quant_enabled_flag u(1)      if( !sps_dep_quant_enabled_flag )
sps_sign_data_hiding_enabled_flag u(1)      sps_virtual_boundaries_enabled_flag u(1)      if(
sps_virtual_boundaries_enabled_flag ) {      sps_virtual_boundaries_present_flag u(1)      if(
sps_virtual_boundaries_present_flag ) {      sps_num_ver_virtual_boundaries u(2)      for( i
= 0; i < sps_num_ver_virtual_boundaries; i++ )      sps_virtual_boundaries_pos_x[ i ] u(13)
    sps_num_hor_virtual_boundaries u(2)      for( i = 0; i <
sps_num_hor_virtual_boundaries; i++ )      sps_virtual_boundaries_pos_y[ i ] u(13)      }
    } if( sps_ptl_dpb_hrd_params_present_flag ) {      sps_general_hrd_params_present_flag
u(1)      if( sps_general_hrd_params_present_flag ) {      general_hrd_parameters( )      if(
sps_max_sublayers_minus1 > 0 )      sps_sublayer_cpb_params_present_flag u(1)
firstSubLayer = sps_sublayer_cpb_params_present_flag ? 0 :
sps_max_sublayers_minus1      ols_hrd_parameters( firstSubLayer,
sps_max_sublayers_minus1 )      } } field_seq_flag u(1)      vui_parameters_present_flag u(1)
    if( vui_parameters_present_flag )      vui_parameters( ) /* Specified in ITU-T H.SEI | ISO/IEC
23002-7 */      sps_extension_flag u(1)      if( sps_extension_flag )      while( more_rbsp_data( ) )
    sps_extension_data_flag u(1)      rbsp_trailing_bits( )      }

```

(27) An SPS table as above may be expanded as described above to incorporate a global motion present indicator as shown in Table 6:

(28) TABLE-US-00006 Descriptor sequence_parameter_set_rbsp() {

sps_sequence_parameter_set_id ue(v) . . . global_motion_present u(1) rbsp_trailing_bits() }

(29) Additional fields may be incorporated in an SPS to reflect further indicators as described in this disclosure.

(30) In an embodiment, and still referring to FIG. 2, an sps_affine_enabled_flag in a PPS and/or SPS may specify whether affine model based motion compensation may be used for inter prediction. If sps_affine_enabled_flag is equal to 0, the syntax may be constrained such that no affine model based motion compensation is used in the code later video sequence (CLVS), and inter affine flag and cu_affine_type_flag may not be present in coding unit syntax of the CLVS. Otherwise (sps_affine_enabled_flag is equal to 1), affine model based motion compensation can be used in the CLVS.

(31) Continuing to refer to FIG. 2, sps_affine_type_flag in a PPS and/or SPS may specify whether 6-parameter affine model based motion compensation may be used for inter prediction. If sps_affine_type_flag is equal to 0, syntax may be constrained such that no 6-parameter affine model based motion compensation is used in the CLVS, and cu_affine_type_flag may not be present in coding unit syntax in the CLVS. Otherwise (sps_affine_type_flag equal to 1), 6-parameter affine model based motion compensation may be used in CLVS. When not present, the value of sps_affine_type_flag may be inferred to be equal to 0.

(32) Still referring to FIG. 2, translational CPMVs may be signaled in a PPS. Control points may be predefined. For example, control point MV 0 may be relative to a top left corner of a picture, MV 1 may be relative to a top right corner, and MV 3 may be relative to a bottom left corner of a picture. Table 4 illustrates an example approach for signaling CPMV data depending on the motion model used.

(33) In an exemplary embodiment, and still referring to FIG. 2, an array mvr_precision_idx, which may be signaled in coding unit, coding tree, or the like, may specify a resolution AmvrShift of a motion vector difference, which may be defined as a non-limiting example as shown in Table 7 as shown below. Array indices x0, y0 may specify the location (x0, y0) of a top-left luma sample of a considered coding block relative to a top-left luma sample of the picture; when amvr_precision_idx[x0][y0] is not present, it may be inferred to be equal to 0. Where an inter affine flag[x0][y0] is equal to 0, variables MvdL0[x0][y0][0], MvdL0[x0][y0][1], MvdL1[x0][y0]

[0], MvdL1[x0][y0][1] representing motion vector difference values corresponding to conserved block, may be modified by shifting such values by AmvrShift, for instance using MvdL0[x0][y0][0]=MvdL0[x0][y0][0]>>AmvrShift; MvdL0[x0][y0][1]=MvdL0[x0][y0][1]<<AmvrShift; MvdL1[x0][y0][0]=MvdL1[x0][y0][0]<<AmvrShift; and MvdL1[x0][y0][1]=MvdL1[x0][y0][1]<<AmvrShift. Where inter affine flag[x0][y0] is equal to 1, variables MvdCpL0[x0][y0][0][0], MvdCpL0[x0][y0][0][1], MvdCpL0[x0][y0][1][0], MvdCpL0[x0][y0][1][1], MvdCpL0[x0][y0][2][0] and MvdCpL0[x0][y0][2][1] may be modified via shifting, for instance as follows: MvdCpL0[x0][y0][0][0]=MvdCpL0[x0][y0][0][0]<<AmvrShift; MvdCpL1[x0][y0][0][1]=MvdCpL1[x0][y0][0][1]<<AmvrShift; MvdCpL0[x0][y0][1][0]=MvdCpL0[x0][y0][1][0]<<AmvrShift; MvdCpL1[x0][y0][1][1]=MvdCpL1[x0][y0][1][1]<<AmvrShift; MvdCpL0[x0][y0][2][0]=MvdCpL0[x0][y0][2][0]<<AmvrShift; and MvdCpL1[x0][y0][2][1]=MvdCpL1[x0][y0][2][1]<<AmvrShift

(34) TABLE-US-00007 AmvrShift inter_affine_flag = 0 CuPredMode[chType] && amvr_precision [x0] CuPredMode[chType][x0] amvr_flag sion_id inter_affine_flag = 1 [y0] = MODE_IBC) [y0] != MODE_IBC 0 — 2 (1/4 luma — 2 (1/4 luma sample) sample) 1 0 0 (1/16 luma 4 (1 luma sample) 3 (1/2 luma sample) sample) 1 1 4 (1 luma sample) 6 (4 luma samples) 4 (1 luma sample) 1 2 — — 6 (4 luma samples)

(35) With continued reference to FIG. 2, global motion may be relative to a previously coded frame. When only one set of global motion parameters are present, motion may be relative to a frame that is presented immediately before current frame.

(36) Still referring to FIG. 2, some implementations of current subject matter may include predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame to improve compression.

(37) Continuing to refer to FIG. 2, a current picture being encoded as an inter picture may use motion estimation to improve compression. Global motion vectors for a current picture may be signaled in a PPS. In some approaches to video compression, when encoding global motion parameters (e.g., control point motion vectors) in a current frame, the following information may already be decoded and available: 1) global motion parameters from a previous frame; 2) global motion parameters relative to available reference pictures in List0 that are already encoded in a current frame; and 3) control point motion vectors in global motion parameters being coded.

(38) Still referring to FIG. 2, predicted motion vector (PMV) of control point motion vectors (CPMV) may be determined from previously coded motion vectors and the difference between the CPMV and PMV may be coded to reduce bits and improve compression efficiency.

(39) For example, and continuing to refer to FIG. 2, CPMV0.sub.i, CPMV1.sub.i, and CPMV2.sub.i may be three control point motion vectors of a frame 'i' to be coded. In an exemplary method, CPMV0.sub.i-1, representing a vector component and/or vector determined for a frame previous to a current frame, including without limitation a reference frame and/or an immediately preceding frame may be used as a prediction of CPMV0.sub.i and a difference between motion vectors may be coded. Difference between x and y component of motion vector and its prediction may be coded.

(40) Still referring to FIG. 2, for CPMV(j,i), j, on a range of 0<=j<3 may be a motion vector number, and i, on a range 0<=i<=ref_pic_count, may be a reference picture index. ref_pic_count=0 may refer to a current picture. CPMV(j,1) may be used as a prediction of CPMV(j,0). Control points for global motion in a frame may be at corners of a frame and CPMV at corresponding corners of frames are likely to be similar and serve as a better prediction.

(41) With continued reference to FIG. 2, and as a non-limiting example, a more complex motion vector prediction may use CPMV for all available reference pictures in list. In this exemplary method, CPMV(j,i) may be used as a prediction of CPMV(j,0). In this case, an index i may also be coded along with motion vector differences.

(42) Still referring to FIG. 2, previously coded CPMV may be used as a prediction to encode a

subsequent CPMV, which may be the next CPMV. For example, CPMV(j,i-1), may be used as a prediction of CPMV(j,i). In this case, an index i may also be coded along with motion vector differences. When only one set CPMV are coded, CPMV0 may be a prediction for CPMV1 and CPMV2.

(43) Further referring to FIG. 2, and as a non-limiting example, table 5 shows an example PPS with global motion parameters using control point motion vectors.

(44) TABLE-US-00008 Descriptor pps_global_motion_parameters_mvd () { motion_model_used u(2) mv0_x se(v) mv1_y se(v) if(motion_model_used == 1){ mv1_x - mv0_x se(v) mv1_y - m01_y se(v) } if(motion_model_used == 2){ mv2_x - mv0_x se(v) mv2_y - m01_y se(v) }

(45) Table 6 shows another example PPS with differentially coded global motion parameters for one or more frames in a reference picture list.

(46) TABLE-US-00009 Descriptor pic_parameter_set_rbsp() { pps_pic_parameter_set_id ue(v) pps_seq_parameter_set_id ue(v) . . . ref_pic_count u(4) for(i=1; i <= ref_pic_count; i++){ gmc_present[i] u(1) } for(i=1; i <= ref_pic_count; i++){ if(gmc_present[i]) mvp = get_cpmv_pred(i); pps_global_motion_parameters_mvp (mvp) } rbsp_trailing bits() }

(47) With continued to reference to FIG. 2, the following is example pseudo code for deriving a predicted CPMV according to an example implementation:

(48) TABLE-US-00010 get_cpmv_pred(i, j){ if(i == 0){ if(j == 0){ pmv.x = 0; pmv.y = 0; }else{ pmv.x = mv0_x; pmv.y = mv0_y; } }else{ pmv[j] = cpmv(j, i-1) return pmv; }

(49) In many instances, and still referring to FIG. 2, global motion may be likely to be present for a set of frames. Global motion may be terminated upon a scene change or when camera motion stops. Accordingly, a global motion of successive frames is likely to be similar. In some implementations, if global motion is used in a previous frame, a CPMV of the previous frame is likely to be a good predictor thereby reducing motion vector difference and bits required to code a motion vector.

(50) In some implementation, and continuing to refer to FIG. 2, it is possible to implicitly code a zero residual for the global motion vector residual by adopting a prior frame's global motion information. For example, if global motion is enabled and if a global motion skip mode (e.g., global_motion_skip_flag) is enabled, then global motion information of a prior frame may be adopted by a current frame as the current frame's global motion. In some implementations, an index may be provided to the reference list (e.g., list0) indicating from which reference frame motion information may be adopted.

(51) FIG. 3 is a process flow diagram illustrating an example process 300 of predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame. At step 305, a current block is received by a decoder. Current block may be contained within a bitstream that a decoder receives. Bitstream may include, for example, data found in a stream of bits that is an input to a decoder when using data compression. Bitstream may include information necessary to decode a video. Receiving may include extracting and/or parsing a block and associated signaling information from bit stream. In some implementations, a current block may include a coding tree unit (CTU), a coding unit (CU), or a prediction unit (PU).

(52) At step 310, and still referring to FIG. 3, a residual of a control point motion vector for a current frame may be extracted from bitstream. At step 315, a residual of the control point motion vector can be combined with a prediction of the control point motion vector for the current frame.

(53) FIG. 4 is a system block diagram illustrating an example decoder 400 capable of decoding a bitstream 428 with predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame. Decoder 400 may include an entropy decoder processor 404, an inverse quantization and inverse transformation processor 408, a deblocking filter 412, a frame buffer 416, motion compensation processor 420 and/or intra prediction processor 424.

(54) In operation, and still referring to FIG. 4, bit stream 428 may be received by a decoder 400 and input to entropy decoder processor 404, which entropy decodes portions of a bit stream into quantized coefficients. Quantized coefficients may be provided to inverse quantization and inverse

transformation processor **408**, which may perform inverse quantization and inverse transformation to create a residual signal, which may be added to the output of motion compensation processor **420** or intra prediction processor **424** according to a processing mode. An output of motion compensation processor **420** and intra prediction processor **424** may include a block prediction based on a previously decoded block. A sum of prediction and residual may be processed by deblocking filter **630** and stored in a frame buffer **640**.

(55) FIG. 5 is a process flow diagram illustrating an exemplary embodiment of a process **200** of encoding a video with predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame according to some aspects of a current subject matter that may reduce encoding complexity while increasing compression efficiency. At step **505**, a video frame may undergo initial block segmentation, for example, using a tree-structured macro block partitioning scheme that may include partitioning a picture frame into CTUs and CUs. At step **510**, a residual of a control point motion vector for a current frame may be determined. At step **515**, a block may be encoded and included in bitstream. Encoding may include utilizing inter prediction and intra prediction modes, for example.

(56) FIG. 6 is a system block diagram illustrating an example video encoder **600** capable of predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame according to some aspects of current subject matter. Example video encoder **600** may receive an input video **604**, which may be initially segmented or dividing according to a processing scheme, such as a tree-structured macro block partitioning scheme (e.g., quad-tree plus binary tree). An example of a tree-structured macro block partitioning scheme may include partitioning a picture frame into large block elements called coding tree units (CTU). In some implementations, each CTU may be further partitioned one or more times into a number of sub-blocks called coding units (CU). A final result of this portioning may include a group of sub-blocks that may be called predictive units (PU). Transform units (TU) may also be utilized.

(57) An example video encoder **600** may include an intra prediction processor **415**, a motion estimation/compensation processor **612** (also referred to as an inter prediction processor) capable of supporting prediction of global motion vectors in a current frame from previously encoded global motion vectors of a previous frame according to some aspects of current subject matter, a transform/quantization processor **616**, an inverse quantization/inverse transform processor **620**, an in-loop filter **624**, a decoded picture buffer **628**, and/or an entropy coding processor **632**. Bit stream parameters may be input to the entropy coding processor **632** for inclusion in the output bit stream **636**.

(58) In operation, and still referring to FIG. 6, for each block of a frame of an input video **604**, whether to process a block via intra picture prediction or using motion estimation/compensation may be determined. A block may be provided to an intra prediction processor **608** or a motion estimation/compensation processor **612**. If block is to be processed via intra prediction, an intra prediction processor **608** may perform processing to output a predictor. If block is to be processed via motion estimation/compensation, a motion estimation/compensation processor **612** can perform processing including predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame according to some aspects of the current subject matter, if applicable.

(59) Further referring to FIG. 6, a residual may be formed by subtracting a predictor from the input video. Residual may be received by transform/quantization processor **616**, which may perform transformation processing (e.g., discrete cosine transform (DCT)) to produce coefficients, which may be quantized. Quantized coefficients and any associated signaling information may be provided to the entropy coding processor **632** for entropy encoding and inclusion in an output bit stream **636**. Entropy encoding processor **632** may support encoding of signaling information related to encoding a current block. In addition, quantized coefficients may be provided to inverse quantization/inverse transformation processor **620**, which may reproduce pixels, which may be

combined with predictor and processed by in loop filter **624**, an output of which may be stored in a decoded picture buffer **628** for use by a motion estimation/compensation processor **612** that is capable of predicting global motion vectors in a current frame from previously encoded global motion vectors of a previous frame according to some aspects of the current subject matter.

(60) Still referring to FIG. **6**, although a few variations have been described in detail above, other modifications or additions are possible. For example, in some implementations, current blocks may include any symmetric blocks (8×8, 16×16, 32×32, 64×64, 128×128, and the like) as well as any asymmetric block (8×4, 16×8, and the like).

(61) In some implementations, and continuing to refer to FIG. **6**, a quadtree plus binary decision tree (QTBT) may be implemented. In QTBT, at a Coding Tree Unit level, partition parameters of QTBT may be dynamically derived to adapt to local characteristics without transmitting any overhead. Subsequently, at Coding Unit level, a joint-classifier decision tree structure may eliminate unnecessary iterations and control the risk of false prediction. In some implementations, LTR frame block update mode may be available as an additional option available at every leaf node of the QTBT.

(62) In some implementations, and still referring to FIG. **6**, additional syntax elements may be signaled at different hierarchy levels of the bitstream. For example, a flag may be enabled for an entire sequence by including an enable flag coded in a Sequence Parameter Set (SPS). Further, a CTU flag may be coded at a coding tree unit (CTU) level.

(63) It is to be noted that any one or more of the aspects and embodiments described herein may be conveniently implemented using digital electronic circuitry, integrated circuitry, specially designed application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs) computer hardware, firmware, software, and/or combinations thereof, as realized and/or implemented in one or more machines (e.g., one or more computing devices that are utilized as a user computing device for an electronic document, one or more server devices, such as a document server, etc.)

programmed according to the teachings of the present specification, as will be apparent to those of ordinary skill in the computer art. These various aspects or features may include implementation in one or more computer programs and/or software that are executable and/or interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device. Appropriate software coding may readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those of ordinary skill in the software art. Aspects and implementations discussed above employing software and/or software modules may also include appropriate hardware for assisting in the implementation of the machine executable instructions of the software and/or software module.

(64) Such software may be a computer program product that employs a machine-readable storage medium. A machine-readable storage medium may be any medium that is capable of storing and/or encoding a sequence of instructions for execution by a machine (e.g., a computing device) and that causes the machine to perform any one of the methodologies and/or embodiments described herein. Examples of a machine-readable storage medium include, but are not limited to, a magnetic disk, an optical disc (e.g., CD, CD-R, DVD, DVD-R, etc.), a magneto-optical disk, a read-only memory “ROM” device, a random access memory “RAM” device, a magnetic card, an optical card, a solid-state memory device, an EPROM, an EEPROM, Programmable Logic Devices (PLDs), and/or any combinations thereof. A machine-readable medium, as used herein, is intended to include a single medium as well as a collection of physically separate media, such as, for example, a collection of compact discs or one or more hard disk drives in combination with a computer memory. As used herein, a machine-readable storage medium does not include transitory forms of signal transmission.

(65) Such software may also include information (e.g., data) carried as a data signal on a data

carrier, such as a carrier wave. For example, machine-executable information may be included as a data-carrying signal embodied in a data carrier in which the signal encodes a sequence of instruction, or portion thereof, for execution by a machine (e.g., a computing device) and any related information (e.g., data structures and data) that causes the machine to perform any one of the methodologies and/or embodiments described herein.

(66) Examples of a computing device include, but are not limited to, an electronic book reading device, a computer workstation, a terminal computer, a server computer, a handheld device (e.g., a tablet computer, a smartphone, etc.), a web appliance, a network router, a network switch, a network bridge, any machine capable of executing a sequence of instructions that specify an action to be taken by that machine, and any combinations thereof. In one example, a computing device may include and/or be included in a kiosk.

(67) FIG. 7 shows a diagrammatic representation of one embodiment of a computing device in the exemplary form of a computer system **700** within which a set of instructions for causing a control system to perform any one or more of the aspects and/or methodologies of the present disclosure may be executed. It is also contemplated that multiple computing devices may be utilized to implement a specially configured set of instructions for causing one or more of the devices to perform any one or more of the aspects and/or methodologies of the present disclosure. Computer system **700** includes a processor **704** and a memory **708** that communicate with each other, and with other components, via a bus **712**. Bus **712** may include any of several types of bus structures including, but not limited to, a memory bus, a memory controller, a peripheral bus, a local bus, and any combinations thereof, using any of a variety of bus architectures.

(68) Memory **708** may include various components (e.g., machine-readable media) including, but not limited to, a random-access memory component, a read only component, and any combinations thereof. In one example, a basic input/output system **716** (BIOS), including basic routines that help to transfer information between elements within computer system **700**, such as during start-up, may be stored in memory **708**. Memory **708** may also include (e.g., stored on one or more machine-readable media) instructions (e.g., software) **720** embodying any one or more of the aspects and/or methodologies of the present disclosure. In another example, memory **708** may further include any number of program modules including, but not limited to, an operating system, one or more application programs, other program modules, program data, and any combinations thereof.

(69) Computer system **700** may also include a storage device **724**. Examples of a storage device (e.g., storage device **724**) include, but are not limited to, a hard disk drive, a magnetic disk drive, an optical disc drive in combination with an optical medium, a solid-state memory device, and any combinations thereof. Storage device **724** may be connected to bus **712** by an appropriate interface (not shown). Example interfaces include, but are not limited to, SCSI, advanced technology attachment (ATA), serial ATA, universal serial bus (USB), IEEE 1394 (FIREWIRE), and any combinations thereof. In one example, storage device **724** (or one or more components thereof) may be removably interfaced with computer system **700** (e.g., via an external port connector (not shown)). Particularly, storage device **724** and an associated machine-readable medium **728** may provide nonvolatile and/or volatile storage of machine-readable instructions, data structures, program modules, and/or other data for computer system **700**. In one example, software **720** may reside, completely or partially, within machine-readable medium **728**. In another example, software **720** may reside, completely or partially, within processor **704**.

(70) Computer system **700** may also include an input device **732**. In one example, a user of computer system **700** may enter commands and/or other information into computer system **700** via input device **732**. Examples of an input device **732** include, but are not limited to, an alpha-numeric input device (e.g., a keyboard), a pointing device, a joystick, a gamepad, an audio input device (e.g., a microphone, a voice response system, etc.), a cursor control device (e.g., a mouse), a touchpad, an optical scanner, a video capture device (e.g., a still camera, a video camera), a touchscreen, and any combinations thereof. Input device **732** may be interfaced to bus **712** via any

of a variety of interfaces (not shown) including, but not limited to, a serial interface, a parallel interface, a game port, a USB interface, a FIREWIRE interface, a direct interface to bus **712**, and any combinations thereof. Input device **732** may include a touch screen interface that may be a part of or separate from display **736**, discussed further below. Input device **732** may be utilized as a user selection device for selecting one or more graphical representations in a graphical interface as described above.

(71) A user may also input commands and/or other information to computer system **700** via storage device **724** (e.g., a removable disk drive, a flash drive, etc.) and/or network interface device **740**. A network interface device, such as network interface device **740**, may be utilized for connecting computer system **700** to one or more of a variety of networks, such as network **744**, and one or more remote devices **748** connected thereto. Examples of a network interface device include, but are not limited to, a network interface card (e.g., a mobile network interface card, a LAN card), a modem, and any combination thereof. Examples of a network include, but are not limited to, a wide area network (e.g., the Internet, an enterprise network), a local area network (e.g., a network associated with an office, a building, a campus or other relatively small geographic space), a telephone network, a data network associated with a telephone/voice provider (e.g., a mobile communications provider data and/or voice network), a direct connection between two computing devices, and any combinations thereof. A network, such as network **744**, may employ a wired and/or a wireless mode of communication. In general, any network topology may be used. Information (e.g., data, software **720**, etc.) may be communicated to and/or from computer system **700** via network interface device **740**.

(72) Computer system **700** may further include a video display adapter **752** for communicating a displayable image to a display device, such as display device **736**. Examples of a display device include, but are not limited to, a liquid crystal display (LCD), a cathode ray tube (CRT), a plasma display, a light emitting diode (LED) display, and any combinations thereof. Display adapter **752** and display device **736** may be utilized in combination with processor **704** to provide graphical representations of aspects of the present disclosure. In addition to a display device, computer system **700** may include one or more other peripheral output devices including, but not limited to, an audio speaker, a printer, and any combinations thereof. Such peripheral output devices may be connected to bus **712** via a peripheral interface **756**. Examples of a peripheral interface include, but are not limited to, a serial port, a USB connection, a FIREWIRE connection, a parallel connection, and any combinations thereof.

(73) The foregoing has been a detailed description of illustrative embodiments of the invention. Various modifications and additions can be made without departing from the spirit and scope of this invention. Features of each of the various embodiments described above may be combined with features of other described embodiments as appropriate in order to provide a multiplicity of feature combinations in associated new embodiments. Furthermore, while the foregoing describes a number of separate embodiments, what has been described herein is merely illustrative of the application of the principles of the present invention. Additionally, although particular methods herein may be illustrated and/or described as being performed in a specific order, the ordering is highly variable within ordinary skill to achieve embodiments as disclosed herein. Accordingly, this description is meant to be taken only by way of example, and not to otherwise limit the scope of this invention.

(74) In the descriptions above and in the claims, phrases such as “at least one of” or “one or more of” may occur followed by a conjunctive list of elements or features. The term “and/or” may also occur in a list of two or more elements or features. Unless otherwise implicitly or explicitly contradicted by the context in which it is used, such a phrase is intended to mean any of the listed elements or features individually or any of the recited elements or features in combination with any of the other recited elements or features. For example, the phrases “at least one of A and B;” “one or more of A and B;” and “A and/or B” are each intended to mean “A alone, B alone, or A and B

together.” A similar interpretation is also intended for lists including three or more items. For example, the phrases “at least one of A, B, and C;” “one or more of A, B, and C;” and “A, B, and/or C” are each intended to mean “A alone, B alone, C alone, A and B together, A and C together, B and C together, or A and B and C together.” In addition, use of the term “based on,” above and in the claims is intended to mean, “based at least in part on,” such that an unrecited feature or element is also permissible.

(75) The subject matter described herein can be embodied in systems, apparatus, methods, and/or articles depending on the desired configuration. The implementations set forth in the foregoing description do not represent all implementations consistent with the subject matter described herein. Instead, they are merely some examples consistent with aspects related to the described subject matter. Although a few variations have been described in detail above, other modifications or additions are possible. In particular, further features and/or variations can be provided in addition to those set forth herein. For example, the implementations described above can be directed to various combinations and sub-combinations of the disclosed features and/or combinations and sub-combinations of several further features disclosed above. In addition, the logic flows depicted in the accompanying figures and/or described herein do not necessarily require the particular order shown, or sequential order, to achieve desirable results. Other implementations may be within the scope of the following claims.

Claims

1. A video encoder configured to encode a bit stream to be decoded by a decoder, the decoder configured to: receive a bit stream, the bitstream including coded picture data for a sequence of coded pictures and motion vector difference information for two control point motion vectors for 4-parameter affine motion compensation; combine the motion vector difference information from the bitstream with a prediction of two control point motion vectors to form two current control point motion vectors, said prediction of two control point motion vectors being obtained from motion vectors utilized to decode a temporally prior picture in the sequence to the current picture; and utilize the two current control point motion vectors and 4-parameter affine motion compensation to decode coded picture data of the current picture in the sequence of coded pictures.
 2. The video encoder of claim 1, wherein the coded picture data further comprises a first contiguous plurality of coding units with common motion and a second contiguous plurality of coding units with local motion.
 3. The video encoder of claim 2, wherein the first contiguous plurality of coding units and the second contiguous plurality of units comprise the entire picture.
 4. The video encoder of claim 3, wherein the coded picture comprises a plurality of $n \times n$ coding units.
 5. The video encoder of claim 4, wherein the value of n is one of 16, 32, 64 or 128.
 6. The video encoder of claim 1, wherein the prediction of two control point motion vectors is obtained from motion vectors utilized to decode a temporally prior picture in the sequence to the current picture.
-