(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118450129 A
(43)申请公布日 2024.08.06

(21)申请号 202410552037.9

(22)申请日 2020.12.18

(30)优先权数据
62/950,453 2019.12.19 US
17/026,748 2020.09.21 US

(62)分案原申请数据
202080035733.2 2020.12.18

(71)申请人 腾讯美国有限责任公司
地址 美国加利福尼亚州帕洛阿尔托公园大道2747号

(72)发明人 李翎 许晓中 崔秉斗 李翔 史蒂芬·文格尔 刘杉

(74)专利代理机构 北京德琦知识产权代理有限公司 11018
专利代理师 程杰 王琦

(51)Int.Cl.
H04N 19/159(2014.01)
H04N 19/174(2014.01)
H04N 19/70(2014.01)

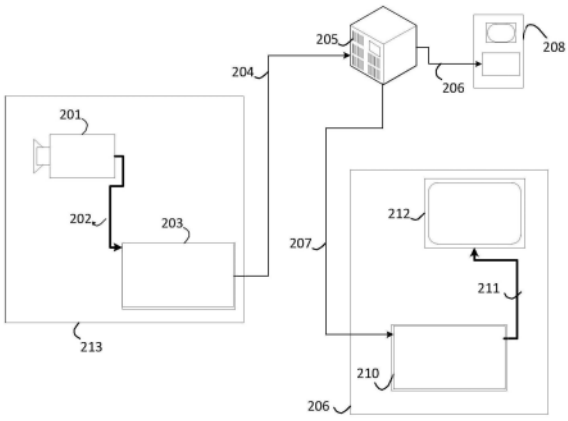
权利要求书2页 说明书54页 附图3页

(54)发明名称

对视频数据进行编码或解码的方法和装置

(57)摘要

本公开实施例提供一种对视频数据进行编码或解码的方法和装置,所述方法包括使用语法元素指示已编码图片的所有切片的切片类型,所述语法元素使用无符号整数进行编码,以及基于通过所述语法元素指示的所述切片类型,对所述视频数据进行编码或解码。



CN 118450129 A

1.一种对视频数据进行解码的方法,其特征在于,所述方法包括:

使用语法元素指示已编码图片的所有切片的切片类型,所述语法元素使用无符号整数进行编码,其中,与图片头相关的语法元素包括在切片层原始字节序列载荷网络抽象层单元中,并且使用标志来指示在所述切片层原始字节序列载荷网络抽象层单元中,存在所述与图片头相关的语法元素;以及

基于通过所述语法元素指示的所述切片类型,对所述视频数据进行解码。

2.根据权利要求1所述的方法,其特征在于,对于所述已编码图片,仅对相关语法元素进行编码。

3.根据权利要求2所述的方法,其特征在于,当所述已编码图片的所有切片被指示为包括帧内预测时,不对帧间预测语法元素进行编码。

4.根据权利要求1所述的方法,其特征在于,所述切片类型从解码的访问单元分隔符值中推断出来。

5.根据权利要求1所述的方法,其特征在于,当所述切片类型在高级语法中发信号通知时,推断所述切片类型。

6.根据权利要求1所述的方法,其特征在于,所述切片类型基于所述已编码图片中的矩形切片的数量推断出来。

7.根据权利要求1-6任一项所述的方法,其特征在于,所述语法元素是0阶指数哥伦布Exp-Golomb编码的语法元素。

8.根据权利要求1所述的方法,其特征在于,所述语法元素是可配置为三种状态的2位语法元素。

9.根据权利要求1所述的方法,其特征在于,所述语法元素是可配置为四种状态的2位语法元素。

10.一种对视频数据进行编码的方法,其特征在于,所述方法包括:

使用语法元素指示已编码图片的所有切片的切片类型,所述语法元素使用无符号整数进行编码,其中,与图片头相关的语法元素包括在切片层原始字节序列载荷网络抽象层单元中,并且使用标志来指示在所述切片层原始字节序列载荷网络抽象层单元中,存在所述与图片头相关的语法元素;以及

基于通过所述语法元素指示的所述切片类型,对所述视频数据进行编码。

11.一种对视频数据进行解码的装置,其特征在于,所述装置包括:

至少一个存储器,被配置为存储计算机程序代码;以及

至少一个处理器,被配置为访问所述至少一个存储器,并根据所述计算机程序代码进行操作,以执行权利要求1-9任一项所述的方法。

12.一种对视频数据进行编码的装置,其特征在于,所述装置包括:

至少一个存储器,被配置为存储计算机程序代码;以及

至少一个处理器,被配置为访问所述至少一个存储器,并根据所述计算机程序代码进行操作,以执行权利要求10所述的方法。

13.一种非易失性计算机可读存储介质,其特征在于,用于存储指令,所述指令使至少一个处理器:以执行权利要求1-10任一项所述的方法。

14.一种存储视频比特流的方法,其特征在于,在非易失性计算机可读存储介质上存储

视频比特流,所述视频比特流根据权利要求10所述的编码方法产生,或者基于权利要求1至9任一项所述的解码方法进行解码。

对视频数据进行编码或解码的方法和装置

[0001] 相关申请的交叉引用

[0002] 本申请要求于2019年12月19日提交的、申请号为62/950,453的美国临时专利申请、以及于2020年9月21日提交的、申请号为17/026,748的美国专利申请的优先权,其全部内容并入本文。

技术领域

[0003] 本公开总体上涉及视频编码/解码,并且总体上描述了超越HEVC(高效视频编解码)的下一代视频编解码技术,例如,多功能视频编解码(VVC)。更具体地,本公开总体上涉及与图片头处理相关的方法和装置。

背景技术

[0004] 提议的VVC草案7包括称为图片头的HLS(高级语法),该HLS包含应用于已编码图片的所有切片的语法元素,例如,以避免在切片头中发信号通知语法元素,这些语法元素被约束为对图片的所有切片具有相同值。

[0005] 图片参数集

[0006] HLS指定可应用于较低级码工具的语法元素。例如,CTU(编码树单元)大小可以在序列级别或SPS(序列参数集)指定,并且通常不会随着图片的不同而改变。典型的HLS包括SPS、PPS(图片参数集)、PH(图片头)、SH(切片头)和APS(自适应参数集)。

[0007] 不同的HLS包括多个级别的应用,使得常用的语法元素不需要重复编码。例如,SPS指定可应用于序列级别的通用语法元素。PH指定可应用于已编码图片的通用语法元素,该已编码图片可由一个或多个切片组成。

[0008] VVC草案7中PPS包含的语法元素描述如下:

[0009] 表1:VVC草案7中PPS包含的语法元素。

[0010]

语法元素	描述符
pic_parameter_set_rbsp( ) {	
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)

[0011]

conf_win_bottom_offset	ue(v)
}	
scaling_window_flag	u(1)
if(scaling_window_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)
mixed_nalu_types_in_pic_flag	u(1)
pps_subpic_id_signalling_present_flag	u(1)
if(pps_subpics_id_signalling_present_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)
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)
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
slice_width_in_tiles_minus1[i]	ue(v)
slice_height_in_tiles_minus1[i]	ue(v)

[0012]

if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0) {	
num_slices_in_tile_minus1[i]	ue(v)
numSlicesInTileMinus1 = num_slices_in_tile_minus1[i]	
for(j = 0; j < numSlicesInTileMinus1; j++)	
slice_height_in_ctu_minus1[i++]	ue(v)
}	
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)
}	
entropy_coding_sync_enabled_flag	u(1)
if(!no_pic_partition_flag || entropy_coding_sync_enabled_flag)	
entry_point_offsets_present_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)
log2_transform_skip_max_size_minus2	ue(v)
cu_qp_delta_enabled_flag	u(1)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset_present_flag	u(1)
if(pps_joint_cbcr_qp_offset_present_flag)	
pps_joint_cbcr_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++) {	

[0013]

cb_qp_offset_list[i]	se(v)
cr_qp_offset_list[i]	se(v)
if(pps_joint_cbcr_qp_offset_present_flag)	
joint_cbcr_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)
}	
}	
constant_slice_header_params_enabled_flag	u(1)
if(constant_slice_header_params_enabled_flag) {	
pps_dep_quant_enabled_idc	u(2)
for(i = 0; i < 2; i++)	
pps_ref_pic_list_sps_idc[i]	u(2)
pps_mvd_l1_zero_idc	u(2)
pps_collocated_from_l0_idc	u(2)
pps_six_minus_max_num_merge_cand_plus1	ue(v)
pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1	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( )	
}	

[0014] 如上面表1所示,num_slices_in_pic_minus1加1指定参考PPS的每个图片中的矩形切片的数量。num_slices_in_pic_minus1的值在0到MaxSlicesPerPicture-1的范围内,包括0和MaxSlicesPerPicture-1。当no_pic_partition_flag等于1时,num_slices_in_

pic_minus1的值可推断为等于0。

[0015] 如上面表1所示,pps_mvd_l1_zero_idc等于0指定语法元素mvd_l1_zero_flag存在于参考PPS的PH中。并且,pps_mvd_l1_zero_idc等于1或2指定mvd_l1_zero_flag不存在于参考PPS的PH中。此外,保留pps_mvd_l1_zero_idc等于3,以供ITU-T|ISO/IEC将来使用。

[0016] 如上面表1所示,pps_collocated_from_l0_idc等于0指定语法元素collocated_from_l0_flag存在于参考PPS的切片的切片头中。并且,pps_collocated_from_l0_idc等于1或2指定语法元素collocated_from_l0_flag不存在于参考PPS的切片的切片头中。此外,保留pps_collocated_from_l0_idc等于3,以供ITU-T|ISO/IEC将来使用。

[0017] 如上面表1所示,pps_six_minus_max_num_merge_cand_plus1等于0指定pic_six_minus_max_num_merge_cand存在于参考PPS的PH中。并且,pps_six_minus_max_num_merge_cand_plus1大于0指定pic_six_minus_max_num_merge_cand不存在于参考PPS的PH中。pps_six_minus_max_num_merge_cand_plus1的值在0到6的范围内,包括(0和6)。

[0018] 如上面表1所示,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1等于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand存在于参考PPS的切片的PH中。并且,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1大于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand不存在于参考PPS的PH中。pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1的值在0到MaxNumMergeCand-1的范围内。

[0019] 切片层原始字节序列载荷(RBSP)

[0020] 切片层RBSP可由切片头和切片数据组成。

[0021] 表2:切片层RBSP

[0022]

语法元素	描述符
slice_layer_rbsp(){	
slice_header()	
slice_data()	
rbsp_slice_trailing_bits()	
}	

[0023] 图片头和切片头

[0024] 在当前图片参考的PPS中编码的语法元素可能在PH和SH中被覆盖,从而在PH中设置了参考PPS的pic_deblocking_filter_override_flag或在SH中设置了参考PPS的slice_deblocking_filter_override_flag。PH中不存在的那些语法元素可能会存在于SH中。例如,当PH中pic_sao_enabled_present_flag的值(指定SAO相关语法元素的存在)为0时,slice_sao_luma_flag和slice_sao_chroma_flag可以在SH中进行编码,以指示亮度和色度上的样本自适应偏移(SAO)的使用。

[0025] 通过使用PH,已经被约束为在图片的所有切片中相同的语法元素可以针对每个图片在PH中传输一次,以避免信令开销,尤其是当图片中有少量切片时。经常随着切片的不同而变化的语法元素仍然可以在SH中传输,以提供灵活性。

[0026] 在VVC草案7中PH和SH包括的语法元素在表3和表5中描述如下。

[0027] 表3:通用切片头语法

[0028]

语法元素	描述符
picture_header_rbsp( ) {	
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	

[0029]

}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	

[0030]

pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
pic_cu_qp_delta_subdiv_intra_slice	ue(v)
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
pic_cu_chroma_qp_offset_subdiv_intra_slice	ue(v)
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)

[0031]

if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	
pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
}	
}	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	

[0032]

if (! pps_dep_quant_enabled_idc)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits( )	
}	

[0033]

slice_header( ) {	
slice_pic_order_cnt_lsb	u(v)
if(subpics_present_flag)	
slice_subpic_id	u(v)
if(rect_slice_flag || NumTilesInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	ue(v)
slice_type	ue(v)
if(!pic_rpl_present_flag &&((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) || sps_idr_rpl_present_flag)) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_sps_flag[i]	u(1)
if(slice_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
slice_poc_lsb_lt[i][j]	u(v)
slice_delta_poc_msb_present_flag[i][j]	u(1)
if(slice_delta_poc_msb_present_flag[i][j])	
slice_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(pic_rpl_present_flag || ((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) || sps_idr_rpl_present_flag)) {	

[0034]

if((slice_type != I && num_ref_entries[0][RplsIdx[0]] > 1) || (slice_type == B && num_ref_entries[1][RplsIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2: 1); i++)	
if(num_ref_entries[i][RplsIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
}	
if(slice_type != I) {	
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(pic_temporal_mvp_enabled_flag) {	
if(slice_type == B && !pps_collocated_from_l0_idc)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && NumRefIdxActive[0] > 1) || (!collocated_from_l0_flag && NumRefIdxActive[1] > 1))	
collocated_ref_idx	ue(v)
}	
if((pps_weighted_pred_flag && slice_type == P) || (pps_weighted_bipred_flag && slice_type == B))	
pred_weight_table( )	
}	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbcr_enabled_flag)	
slice_joint_cbcr_qp_offset	se(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(sps_sao_enabled_flag && !pic_sao_enabled_present_flag) {	
slice_sao_luma_flag	u(1)

[0035]

if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
if(deblocking_filter_override_enabled_flag && !pic_deblocking_filter_override_present_flag)	
slice_deblocking_filter_override_flag	u(1)
if(slice_deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

[0036] 如上面和下面所述,slice_type可以根据下表4指定切片的编码类型:

[0037] 表4:slice_type

[0038]

slice_type	slice_type的名称
0	B(B slice)

1	P(P slice)
2	I(I slice)

[0039] 访问单元分隔符

[0040] AU(访问单元)分隔符用于指示AU的开始以及AU中的已编码图片中存在的切片的类型,该AU包含AU分隔符NAL(网络抽象层)单元。目前,没有与AU分隔符相关的规范解码过程。

[0041] 并且,pic_type表示包含AU分隔符NAL单元的AU中的已编码图片的所有切片的slice_type值是表4中针对pic_type的给定值列出的集合的成员。在比特流中,pic_type的值可以等于0、1或2。保留pic_type的其它值,以供ITU-T|ISO/IEC将来使用。符合此版本的解码器可以忽略pic_type的保留值。

[0042] 表5:pic_type的解释

[0043]

pic_type	存在于AU中的slice_type值
0	I
1	P,I
2	B,P,I

[0044] 非专利文献[1](“NPL 1”)提出了一种高级控制标志,以指示被覆盖的低级编码层需要一组参数。

[0045] NPL 1描述了一种方法,其中,所有与帧间预测相关的语法元素或参数仅当存在至少一个帧间编码切片时或者仅当图片内存在子分区时,才需要发信号通知。否则,不发信号通知这些语法元素或参数。

[0046] 在NPL 1中描述的一个实施例中,发信号通知图片头中的控制标志(称为pic_intra_only_flag),以指示图片内的所有切片(或该图片的任何种类的子分区)是否将只有帧内预测(或非帧间相关的预测)。当该标志为真时,只有与帧内编码相关的语法元素或参数稍后在图片头中发信号通知。否则,当该标志为假时,发信号通知与帧间预测相关的语法元素或参数。下面提供了反映本实施例的语法表:

[0047] 表6:NPL 1的第一实施例

[0048]

语法元素	描述符
pic_intra_only_flag	u(1)
if(!pic_intra_only_flag){	
if(sps_temporal_mvp_enabled_flag && !pps_temporal_mvp_enabled_idc)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	

[0049]

pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag && !pps_five_minus_max_num_subblock_merge_cand_plus1)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_dmvr_slice_present_flag)	
pic_disable_bdof_dmvr_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	

[0050] 在NPL 1的另一种方法中,当不存在帧间编码切片或当图片内存在子分区时,需要发信号通知仅用于帧内切片或帧内子分区的所有相关的语法元素或参数。否则,不发信号通知这些语法元素或参数。

[0051] 在NPL 1的另一个实施例中,发信号通知图片头中的控制标志(称为pic_inter_only_flag),以指示图片内的所有切片(或该图片的任何种类的子分区)是否将具有帧间预测(或非帧内相关的预测)。当该标志为真时,与帧内切片相关的语法元素或参数稍后不在图片头中发信号通知。否则,当该标志为假时,可以在图片中的一个或多个切片或一个或多个子分区中的至少一个中使用帧内切片。将发信号通知帧内切片或子分区的相关语法元素或参数。下面提供了反映本实施例的语法表:

[0052] 表7:NPL 1的第二实施例

[0053]

语法元素	描述符
pic_inter_only_flag	u(1)
if(!pic_inter_only_flag){	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_chroma	ue(v)
pic_max_mtt_hierarchy_depth_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_chroma	ue(v)

[0054]

pic_log2_diff_max_tt_min_qt_chroma	ue(v)
}	
}	
}	

[0055] 在NPL 1中描述的上述方法中,如果图片有自己的类型,例如为帧内图片或帧间图片,则不需要发信号通知上述控制标志pic_intra_only_flag和pic_inter_only_flag,它们的值可以从图片类型中推导出来。

[0056] 并且,如果当前图片的图片类型为仅帧内图片(图片中的所有切片均为I切片),则pic_intra_only_flag可推断为真。在另一示例中,如果当前图片的图片类型为仅帧间图片(图片中的所有切片均为P或B切片),则pic_inter_only_flag可推断为真。在NPL 1中的又一示例中,如果当前图片的图片类型指示图片中的帧内切片和帧间切片均可能,则pic_intra_only_flag和pic_inter_only_flag均可推断为假。

[0057] 问题

[0058] 尽管可以每个图片发信号通知一次PH,以避免发信号通知图片内的多个切片所共有的语法元素,但是该信令在不考虑仅用于帧内切片(I切片)或帧间切片(B、P切片)的语法元素的情况下,可能会引入开销。

发明内容

[0059] 实施例涉及视频编码/解码的方法、系统和计算机可读介质,更具体地,涉及图片头处理。

附图说明

[0060] 这些和其他目的、特征和优点将从下面结合附图阅读的示例性实施例的详细描述中变得明显。附图的各种特征不是按比例绘制的,因为这些附图是为了便于本领域技术人员结合详细描述进行清楚的理解。在附图中:

[0061] 图1是根据实施例的通信系统的简化框图的示意图。

[0062] 图2是根据实施例的通信系统的简化框图的示意图。

[0063] 图3是根据实施例的计算机系统的示意图。

具体实施方式

[0064] 图1示出了根据本公开实施例的通信系统(100)的简化框图。通信系统(100)可以包括通过网络(150)互联的至少两个终端装置(110-120)。对于单向数据传输,第一终端装置(110)可在本地位置对视频数据进行编码,以通过网络(150)传输到另一终端装置(120)。第二终端装置(120)可从网络(150)接收另一终端装置的已编码视频数据,对已编码视频数据进行解码并显示恢复的视频数据。单向数据传输在媒体服务等应用中是较常见的。

[0065] 图1示出了支持已编码视频的双向传输的第二对终端装置(130,140),所述双向传输可例如在视频会议期间发生。对于双向数据传输,每个终端装置(130,140)可对在本地位置采集的视频数据进行编码,以通过网络(150)传输到另一终端装置。每个终端装置(130,

140)还可接收由另一终端装置传输的已编码视频数据,且可对所述已编码视频数据进行解码并在本地显示设备上显示恢复的视频数据。

[0066] 在图1中,终端装置(110-140)可为服务器、个人计算机和智能手机,但本公开的原理可不限于此。本公开的实施例适用于膝上型计算机、平板电脑、媒体播放器和/或专用视频会议设备。网络(150)表示在终端装置(110-140)之间传送已编码视频数据的任何数目的网络,包括例如有线和/或无线通信网络。通信网络(150)可在电路交换和/或分组交换信道中交换数据。代表性网络包括电信网络、局域网、广域网和/或互联网。出于本讨论的目的,除非在下文中有所解释,否则网络(150)的架构和拓扑对于本公开的操作来说可能是无关紧要的。

[0067] 作为所公开主题的应用实施例,图2示出了视频解码器和编码器在流式传输环境中的放置方式。所公开的主题可同等地适用于其它支持视频的应用,包括例如视频会议、数字TV、在包括CD、DVD、存储棒等的数字介质上存储压缩视频等等。

[0068] 流式传输系统可包括采集子系统(213),所述采集子系统可包括诸如数码相机等视频源(201),所述视频源创建例如未压缩的视频样本流(202)。较于已编码的视频比特流,样本流(202)被描绘为粗线,以强调其为高数据量的视频样本流,样本流(202)可由耦接到相机(201)的编码器(203)处理。编码器(203)可包括硬件、软件或软硬件组合以实现或实施如下文更详细地描述的所公开主题的各方面。相较于样本流(202),已编码的视频比特流(204)被描绘为细线,以强调较低数据量的已编码的视频比特流,其可存储在流式传输服务器(205)上以供将来使用。一个或多个流式传输客户端(206,208)可访问流式传输服务器(205)以检索已编码的视频比特流(204)的副本(207,209)。客户端(206)可包括视频解码器(210)。视频解码器(210)对已编码的视频比特流的传入副本(207)进行解码,且产生可在显示器(212)或另一呈现装置(未示出)上呈现的输出视频样本流(211)。在一些流式传输系统中,可以根据某些视频编解码/压缩标准对视频比特流(204,207,209)进行编码。这些标准的示例包括ITU-T H.265建议书。正在开发的视频编解码标准被非正式地称为多功能视频编解码(VVC)。所公开的主题可以在VVC的上下文中使用。

[0069] 在实施例中,语法元素pic_type_idc可用于指示已编码图片的所有切片的切片类型。

[0070] 在一个实施例中,pic_type_idc可以使用无符号整数0阶指数哥伦布(Exp-Golomb)编码的语法元素进行编码,其中左比特在前。这里,pic_type_idc可以有三个值0、1和2,以及只有三种状态,例如仅I切片、B、P、I切片、以及B、P切片等。该值可以以任何顺序映射到状态。下面的表8示出了可能的pic_type_idc语义的示例。

[0071] 表8:可能的pic_type_idc语义的示例

[0072]

pic_type_idc	已编码图片中存在的slice_type
0	B,P,I
1	I
2	B,P

[0073] 在实施例中,pic_type_idc可以使用使用2位的无符号整数进行编码。这里,pic_type_idc可以有但不一定限于三个值0、1和2,以及三种状态,例如:仅I切片、B、P、I切片、以及B、P切片。可以保留pic_type_idc的其他值,以供进一步定义。

[0074] 表9-可能的pic_type_idc语义的示例

[0075]

pic_type_idc	已编码图片中存在的slice_type
0	B,P,I
1	I
2	B,P
3	reserved

[0076] 在实施例中,保留的pic_type_idc的值3可以仅指示已编码图片中存在的P、I片。

[0077] 在示例中,pic_type_idc可以使用2位的无符号整数进行编码。并且,pic_type_idc可以有四个值0、1、2和3,以及四种状态,例如仅I切片、B、P、I切片、B切片、以及P切片。

[0078] 表10-可能的pic_type_idc语义的示例

[0079]

pic_type_idc	已编码图片中存在的slice_type
0	B,P,I
1	I
2	B
3	P

[0080] 建议在HLS中发信号通知pic_type_idc,以便只编码或显示相关的语法元素,从而减少信令开销。例如,当指示图片的pic_type_idc仅为帧内时,不发信号通知帧间相关的语法元素。

[0081] 在一个示例中,可以在PPS中发信号通知pic_type_idc,使得其为参考PPS的每个已编码图片的所有切片指定切片类型。下面给出了详细的语法和语义。在下表以及本公开的其他表格中,与VVC草案7相比的变化用斜体表示。

[0082] 表11:详细的语法和语义

[0083]

语法元素	描述符
pic_parameter_set_rbsp( ) {	
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
pic_type_idc	*ue(v)*
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
scaling_window_flag	u(1)

[0084]

if(scaling_window_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)
mixed_nalu_types_in_pic_flag	u(1)
pps_subpic_id_signalling_present_flag	u(1)
if(pps_subpics_id_signalling_present_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)
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)
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
slice_width_in_tiles_minus1[i]	ue(v)
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0) {	

[0085]

num_slices_in_tile_minus1[i]	ue(v)
numSlicesInTileMinus1 = num_slices_in_tile_minus1[i]	
for(j = 0; j < numSlicesInTileMinus1; j++)	
slice_height_in_ctu_minus1[i++]	ue(v)
}	
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)
}	
entropy_coding_sync_enabled_flag	u(1)
if(!no_pic_partition_flag || entropy_coding_sync_enabled_flag)	
entry_point_offsets_present_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)
log2_transform_skip_max_size_minus2	ue(v)
cu_qp_delta_enabled_flag	u(1)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset_present_flag	u(1)
if(pps_joint_cbcr_qp_offset_present_flag)	
pps_joint_cbcr_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_cbcr_qp_offset_present_flag)	

[0086]

joint_cbcr_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)
}	
}	
constant_slice_header_params_enabled_flag	u(1)
if(constant_slice_header_params_enabled_flag) {	
pps_dep_quant_enabled_idc	u(2)
for(i = 0; i < 2; i++)	
pps_ref_pic_list_sps_idc[i]	u(2)
if (pic_type_idc !=1) {	
pps_mvd_l1_zero_idc	u(2)
pps_collocated_from_l0_idc	u(2)
pps_six_minus_max_num_merge_cand_plus1	ue(v)
pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1	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( )	
}	

[0087] 这里,pic_type_idc为参考PPS的每个已编码图片的所有切片指定切片类型。

[0088] 在一个实施例中,pic_type_idc设置为1表示参考PPS的每个已编码图片只有一个或多个I切片。在这种情况下,与语法元素pps_mvd_l1_zero_idc、pps_collocated_from_l0_idc、pps_six_minus_max_num_merge_cand_plus1和pps_max_num_merge_cand_minus_

max_num_triangle_cand_plus1相关的帧间切片(B、P切片)可推断为等于0。

[0089] 这里,pps_mvd_l1_zero_idc等于0指定语法元素mvd_l1_zero_flag存在于参考PPS的PH中。并且,pps_mvd_l1_zero_idc等于1或2指定mvd_l1_zero_flag不存在于参考PPS的PH中。此外,保留pps_mvd_l1_zero_idc等于3,以供ITU-T|ISO/IEC将来使用。当不存在时,pps_mvd_l1_zero_idc可推断为0。

[0090] 此外,pps_collocated_from_l0_idc等于0指定语法元素collocated_from_l0_flag存在于参考PPS的切片的切片头中。并且,pps_collocated_from_l0_idc等于1或2指定语法元素collocated_from_l0_flag不存在于参考PPS的切片的切片头中。此外,保留pps_collocated_from_l0_idc等于3,以供ITU-T|ISO/IEC将来使用。当不存在时,pps_collocated_from_l0_idc可推断为等于0。

[0091] 并且,pps_six_minus_max_num_merge_cand_plus1等于0指定pic_six_minus_max_num_merge_cand存在于参考PPS的PH中。此外,pps_six_minus_max_num_merge_cand_plus1大于0指定pic_six_minus_max_num_merge_cand不存在于参考PPS的PH中。pps_six_minus_max_num_merge_cand_plus1的值在0到6的范围内,包括0和6。当不存在时,pps_six_minus_max_num_merge_cand_plus1可推断为等于0。

[0092] 如图所示,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1等于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand存在于参考PPS的切片的PH中。并且,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1大于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand不存在于参考PPS的PH中。pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1的值在0到MaxNumMergeCand-1的范围内。当不存在时,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1可推断为等于0。

[0093] 图12:建议的图片头RBSP语法

[0094]

语法元素	描述符
picture_header_rbsp( ) {	
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)

[0095]

}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)

[0096]

if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
if (pic_type_idc !=2) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	*ue(v)*

[0097]

pic_log2_diff_max_tt_min_qt_inter_slice	*ue(v)*
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
if (pic_type_idc !=1) {	
pic_log2_diff_min_qt_min_cb_inter_slice	
pic_max_mtt_hierarchy_depth_inter_slice	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
if (pic_type_idc !=2)	
pic_cu_qp_delta_subdiv_intra_slice	*ue(v)*
if (pic_type_idc !=1)	
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
if (pic_type_idc !=2)	
pic_cu_chroma_qp_offset_subdiv_intra_slice	*ue(v)*
if (pic_type_idc !=1)	
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)

[0098]

}	
if (pic_type_idc !=1) {	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	
pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
}	
}	

[0099]

if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if (! pps_dep_quant_enabled_idc)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)

[0100]

}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits( )	
}	

[0101] 对于参考PPS的每个已编码图片,pic_type_idc用于确定是否解析与帧内切片(I切片)和帧间切片(B、P切片)相关的语法元素。例如,仅当只有与PH相关联的I切片时,才解码与帧内切片相关的语法元素pic_log2_diff_min_qt_min_cb_intra_slice_luma、pic_max_mtt_hierarchy_depth_intra_slice_luma、

[0102] pic_log2_diff_max_bt_min_qt_intra_slice_luma、

[0103] pic_log2_diff_max_tt_min_qt_intra_slice_luma、

[0104] pic_log2_diff_min_qt_min_cb_intra_slice_chroma、

[0105] pic_max_mtt_hierarchy_depth_intra_slice_chroma、

[0106] pic_log2_diff_max_bt_min_qt_intra_slice_chroma和

[0107] pic_log2_diff_max_tt_min_qt_intra_slice_chroma。另一方面,只要存在帧间切片,就解码与帧间切片相关的语法元素。

[0108] 在一个示例中,在PH中发信号通知pic_type_idc,使得其为与PH相关联的已编码图片的所有切片指定切片类型。详细的语法和语义如下所示。与VVC草案7相比的变化用斜体表示。

[0109] 表13:建议的图片头RBSP语法

[0110]

语法元素	描述符
picture_header_rbsp( ) {	
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
pic_type_idc	*ue(v)*

[0111]

if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0	

[0112]

&& !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
if (pic_type_idc !=2) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	

[0113]

if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	*ue(v)*
pic_log2_diff_max_tt_min_qt_inter_slice	*ue(v)*
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
if (pic_type_idc !=1) {	
pic_log2_diff_min_qt_min_cb_inter_slice	
pic_max_mtt_hierarchy_depth_inter_slice	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
if (pic_type_idc !=2)	
pic_cu_qp_delta_subdiv_intra_slice	*ue(v)*
if (pic_type_idc !=1)	
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
if (pic_type_idc !=2)	

[0114]

pic_cu_chroma_qp_offset_subdiv_intra_slice	*ue(v)*
if (pic_type_idc !=1)	
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if (pic_type_idc !=1) {	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	
pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	

[0115]

pic_sao_chroma_enabled_flag	u(1)
}	
}	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if (! pps_dep_quant_enabled_idc)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	

[0116]

pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits( )	
}	

[0117] 对于每个已编码图片,pic_type_idc用于确定是否解析与帧内切片(I切片)和帧间切片(B、P切片)相关的语法元素。例如,仅当只有与PH相关联的I切片时,才解码与帧内切片相关的语法元素pic_log2_diff_min_qt_min_cb_intra_slice_luma、pic_max_mtt_hierarchy_depth_intra_slice_luma、

[0118] pic_log2_diff_max_bt_min_qt_intra_slice_luma、

[0119] pic_log2_diff_max_tt_min_qt_intra_slice_luma、

[0120] pic_log2_diff_min_qt_min_cb_intra_slice_chroma、

[0121] pic_max_mtt_hierarchy_depth_intra_slice_chroma、

[0122] pic_log2_diff_max_bt_min_qt_intra_slice_chroma和

[0123] pic_log2_diff_max_tt_min_qt_intra_slice_chroma。另一方面,只要存在帧间切片,就解码与帧间切片相关的语法元素。

[0124] 在一个实施例中,pic_type_idc可以存在于PPS和PH中,分别称为pps_pic_type_idc和ph_pic_type_idc。

[0125] 表14:建议的图片参数集RBSP语法

[0126]

语法元素	描述符
pic_parameter_set_rbsp( ) {	
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	u(4)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
pps_pic_type_idc	*ue(v)*

[0127]

conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
scaling_window_flag	u(1)
if(scaling_window_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)
mixed_nalu_types_in_pic_flag	u(1)
pps_subpic_id_signalling_present_flag	u(1)
if(pps_subpics_id_signalling_present_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)
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)

[0128]

tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
slice_width_in_tiles_minus1[i]	ue(v)
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0) {	
num_slices_in_tile_minus1[i]	ue(v)
numSlicesInTileMinus1 = num_slices_in_tile_minus1[i]	
for(j = 0; j < numSlicesInTileMinus1; j++)	
slice_height_in_ctu_minus1[i++]	ue(v)
}	
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)
}	
entropy_coding_sync_enabled_flag	u(1)
if(!no_pic_partition_flag || entropy_coding_sync_enabled_flag)	
entry_point_offsets_present_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)
log2_transform_skip_max_size_minus2	ue(v)
cu_qp_delta_enabled_flag	u(1)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_joint_cbcr_qp_offset_present_flag	u(1)
if(pps_joint_cbcr_qp_offset_present_flag)	
pps_joint_cbcr_qp_offset_value	se(v)

[0129]

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_cbcr_qp_offset_present_flag)	
joint_cbcr_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)
}	
}	
constant_slice_header_params_enabled_flag	u(1)
if(constant_slice_header_params_enabled_flag) {	
pps_dep_quant_enabled_idc	u(2)
for(i = 0; i < 2; i++)	
pps_ref_pic_list_sps_idc[i]	u(2)
if (pps_pic_type_idc !=1) {	
pps_mvd_l1_zero_idc	u(2)
pps_collocated_from_l0_idc	u(2)
pps_six_minus_max_num_merge_cand_plus1	ue(v)
pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1	ue(v)
}	
}	
picture_header_extension_present_flag	u(1)

[0130]

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( )	
}	

[0131] 这里,pps_pic_type_idc为参考PPS的每个已编码图片的所有切片指定切片类型。

[0132] 并且,pps_mvd_l1_zero_idc等于0指定语法元素mvd_l1_zero_flag存在于参考PPS的PH中。此外,pps_mvd_l1_zero_idc等于1或2指定mvd_l1_zero_flag不存在于参考PPS的PH中。此外,保留pps_mvd_l1_zero_idc等于3,以供ITU-T|ISO/IEC将来使用。当不存在时,pps_mvd_l1_zero_idc可推断为等于0。

[0133] 并且,pps_collocated_from_l0_idc等于0指定语法元素collocated_from_l0_flag存在于参考PPS的切片的切片头中。此外,pps_collocated_from_l0_idc等于1或2指定语法元素collocated_from_l0_flag不存在于参考PPS的切片的切片头中。此外,保留pps_collocated_from_l0_idc等于3,以供ITU-T|ISO/IEC将来使用。当不存在时,pps_collocated_from_l0_idc可推断为等于0。

[0134] 并且,pps_six_minus_max_num_merge_cand_plus1等于0指定pic_six_minus_max_num_merge_cand存在于参考PPS的PH中。此外,pps_six_minus_max_num_merge_cand_plus1大于0指定pic_six_minus_max_num_merge_cand不存在于参考PPS的PH中。pps_six_minus_max_num_merge_cand_plus1的值应在0到6的范围内,包括0和6。当不存在时,pps_six_minus_max_num_merge_cand_plus1可推断为等于0。

[0135] 并且,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1等于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand存在于参考PPS的切片的PH中。此外,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1大于0指定pic_max_num_merge_cand_minus_max_num_triangle_cand不存在于参考PPS的PH中。pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1的值应在0到MaxNumMergeCand-1的范围内。当不存在时,pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1可推断为等于0。

[0136] 在一个示例中,当pps_pic_type_idc的值表示存在一种类型的切片(如表10中的值1、2和3的I或B或P切片)时,可以从pps_pic_type_idc的值推断出ph_pic_type_idc的值。

[0137] 表15:建议的图片头RBSP语法

[0138]

语法元素	描述符
picture_header_rbsp( ) {	
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)

[0139]

no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
ph_pic_type_idc	*ue(v)*
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	

[0140]

pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
if (ph_pic_type_idc !=2) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_log2_diff_min_qt_min_cb_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)

[0141]

if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	*ue(v)*
pic_log2_diff_max_tt_min_qt_inter_slice	*ue(v)*
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
if (ph_pic_type_idc !=1) {	
pic_log2_diff_min_qt_min_cb_inter_slice	
pic_max_mtt_hierarchy_depth_inter_slice	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	
if (ph_pic_type_idc !=2)	

[0142]

pic_cu_qp_delta_subdiv_intra_slice	*ue(v)*
if (ph_pic_type_idc !=1)	
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
if (ph_pic_type_idc !=2)	
pic_cu_chroma_qp_offset_subdiv_intra_slice	*ue(v)*
if (ph_pic_type_idc !=1)	
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if (ph_pic_type_idc !=1) {	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	

[0143]

pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
}	
}	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if (! pps_dep_quant_enabled_idc)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	

[0144]

}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits( )	
}	

[0145] 这里,ph_pic_type_idc为与PH相关联的每个已编码图片的所有切片指定切片类型。

[0146] 在一个实施例中,ph_pic_type_idc等于1表示与PH相关联的每个已编码图片只有一个或多个I切片。

[0147] 表16:可能的pic_type_idc语义的示例

[0148]

ph_pic_type_idc	已编码图片中存在的slice_type
0	B,P,I
1	I
2	B,P

[0149] 如果pps_pic_type_idc等于0(如表10中的B、P、I切片),则ph_pic_type_idc的值具有0到2的范围,包括0和2。否则,可以从pps_pic_type_idc推断出ph_pic_type_idc的值(例如,相同的值)。在这种情况下,比特流一致性的要求是ph_pic_type_idc的值等于pps_pic_type_idc的值。

[0150] 在一个示例中,语法ph_pic_type_idc的信令取决于(例如,受约束于)pps_pic_type_idc的值。当pps_pic_type_idc的值指示在已编码图片中同时存在帧内切片(I切片)和帧间切片(B、P切片)时,可能需要发信号通知/解析ph_pic_type_idc,以指示与图片头相

关联的图片中存在的切片类型。在其它情况下,当pps_pic_type_idc指示仅存在一个切片类型时,不发信号通知/解析ph_pic_type_idc,并推断其等于pps_pic_type_idc的切片类型(例如,具有相同切片类型)。比特流一致性的要求是ph_pic_type_idc的范围不大于pps_pic_type_idc的范围。

[0151] 表17:建议的图片都RBSP语法

[0152]

语法元素	描述符
picture_header_rbsp( ) {	
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if (pps_pic_type_idc==0)	
ph_pic_type_idc	*ue(v)*
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
ph_subpic_id_signalling_present_flag	u(1)
if(ph_subpics_id_signalling_present_flag) {	
ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_loop_filter_across_virtual_boundaries_disabled_present_flag	u(1)
if(ph_loop_filter_across_virtual_boundaries_disabled_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)

[0153]

ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 || (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplsIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplsIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(partition_constraints_override_flag) {	
if (ph_pic_type_idc !=2) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)

[0154]

pic_log2_diff_min_qt_min_cb_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_inter_slice	*ue(v)*
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	*ue(v)*
pic_log2_diff_max_tt_min_qt_inter_slice	*ue(v)*
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
if (ph_pic_type_idc !=1) {	
pic_log2_diff_min_qt_min_cb_inter_slice	
pic_max_mtt_hierarchy_depth_inter_slice	
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
}	
}	
if(cu_qp_delta_enabled_flag) {	

[0155]

if (ph_pic_type_idc !=2)	
pic_cu_qp_delta_subdiv_intra_slice	*ue(v)*
if (ph_pic_type_idc !=1)	
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
if (ph_pic_type_idc !=2)	
pic_cu_chroma_qp_offset_subdiv_intra_slice	*ue(v)*
if (ph_pic_type_idc !=1)	
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
}	
if (ph_pic_type_idc !=1) {	
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
pic_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
if (sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbcr_enabled_flag)	

[0156]

pic_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag) {	
pic_sao_enabled_present_flag	u(1)
if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
}	
}	
if(sps_alf_enabled_flag) {	
pic_alf_enabled_present_flag	u(1)
if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
}	
if (! pps_dep_quant_enabled_idc)	
pic_dep_quant_enabled_flag	u(1)
if(!pic_dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag) {	
pic_deblocking_filter_override_present_flag	u(1)
if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	

[0157]

}	
}	
}	
if(sps_lmcs_enabled_flag) {	
pic_lmcs_enabled_flag	u(1)
if(pic_lmcs_enabled_flag) {	
pic_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
pic_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
pic_scaling_list_present_flag	u(1)
if(pic_scaling_list_present_flag)	
pic_scaling_list_aps_id	u(3)
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
rbsp_trailing_bits( )	
}	

[0158] 这里,ph_pic_type_idc为与PH相关联的每个已编码图片的所有切片指定切片类型。并且,ph_pic_type_idc仅在pps_pic_type_idc等于0时才可以存在于比特流中。

[0159] 此外,ph_pic_type_idc等于1表示与PH相关联的每个已编码图片只有一个或多个I切片。如果pps_pic_type_idc等于0(如表8中的B、P、I切片),则ph_pic_type_idc的值具有0到2的范围,包括0和2。否则,当ph_pic_type_idc不存在时,推断其等于pps_ph_type_idc,如表8所示。

[0160] 在一个实施例中,与PH相关的语法元素包括在切片层RBSP NAL单元中,并且ph_present_flag用于指示在切片层RBSP NAL单元中存在与PH相关的语法。重复与PH相关的语法信令可具有错误恢复(error resilience)和错误修复(error recovery)的优点。当PH NAL单元在任何类型的网络中进行传输期间被损坏时,切片层RBSP NAL单元能够因为切片层RBSP NAL单元中存在PH而从错误中恢复。与VVC草案7相比的变化用斜体表示。

[0161] 表18:建议的切片层RBSP语法

[0162]

语法元素	描述符
slice_layer_rbsp( ) {	

[0163]

ph_present_flag	*u(1)*
if (ph_present_flag)	
picture_header_rbsp( )	
slice_header( )	
slice_data( )	
rbsp_slice_trailing_bits( )	
}	

[0164] 这里,ph_present_flag可用于指定切片层RBSP中存在与PH相关的语法。当ph_present_flag等于1时,存在与PH相关的语法。当ph_present_flag等于0时,切片层RBSP中不存在与PH相关的语法。

[0165] 在一个实施例中,当存在如上所述的在AU分隔符中解码的pic_type时,在HLS中发信号通知的pic_type_idc可以从pic_type的值推断出或受其约束。

[0166] 在一个示例中,当如表5所示pic_type等于0时,其指示I切片,比特流一致性的要求是pic_type_idc的值指定每个图片中只有帧内切片。例如,当pic_type_idc符合1时,只有帧内切片。

[0167] 在一个示例中,当pic_type_idc受pic_type的值约束时,pic_type_idc值的范围可以取决于pic_type的值。例如,pic_type_idc的值如表10所述,如果pic_type等于1,则pic_type_idc的值可以为1或3。在其它情况下,当pic_type等于2时,pic_type_idc的值在0到3的范围内。

[0168] 在一个实施例中,当按照上述方法在HLS中发信号通知pic_type_idc时,可以推断slice_type。

[0169] 在一个示例中,当pic_type_idc具有指示只有帧内切片的值时,slice_type可以推断为2。

[0170] 在一个示例中,当pic_type_idc具有指示只有帧间切片的值时,slice_type的值具有0到1的范围,包括0和1。例如,当pic_type_idc的值为2(B、P切片)时,那么slice_type的可能值为0和1。

[0171] 在一个实施例中,slice_type的值可以从pic_type_idc和num_slices_in_pic_minus1的值推断出来。

[0172] 当pic_type_idc的值指示同时存在帧内切片和帧间切片时,比特流一致性的要求是num_slices_in_pic_minus1的值大于或等于1。

[0173] 可能存在如下情况:pic_type_idc的值指示在已编码图片中同时存在帧内切片和帧间切片,并且num_slices_in_pic_minus1的值大于或等于1。

[0174] 当所有的先前已编码切片均为帧间切片时,最后一个切片可以是帧内切片,其slice_type等于2(I切片)。

[0175] 当所有的先前已编码切片均为帧内切片时,最后一个切片可以是帧间切片,其slice_type值的范围为0到1,包括0和1。

[0176] 上面提出的方法可以通过处理电路(例如,一个或多个处理器或一个或多个集成电路)来实现。在一个示例中,一个或多个处理器执行存储在非易失性计算机可读介质中的程序,以执行所提出的方法中的一个或多个。

[0177] 上述技术可以通过计算机可读指令实现为计算机软件,并且物理地存储在一个或多个计算机可读介质中。例如,图3示出了计算机系统300,其适于实现所公开主题的某些实施例。

[0178] 所述计算机软件可通过任何合适的机器代码或计算机语言进行编码,通过汇编、编译、链接等机制创建包括指令的代码,所述指令可由计算机中央处理单元(CPU),图形处理单元(GPU)等直接执行或通过译码、微代码等方式执行。

[0179] 所述指令可以在各种类型的计算机或其组件上执行,包括例如个人计算机、平板电脑、服务器、智能手机、游戏设备、物联网设备等。

[0180] 图3所示的用于计算机系统300的组件本质上是示例性的,并不用于对实现本公开实施例的计算机软件的使用范围或功能进行任何限制。也不应将组件的配置解释为与计算机系统300的示例性实施例中所示的任一组件或其组合具有任何依赖性或要求。

[0181] 计算机系统300可以包括某些人机界面输入设备。这种人机界面输入设备可以通过触觉输入(如:键盘输入、滑动、数据手套移动)、音频输入(如:声音、掌声)、视觉输入(如:手势)、嗅觉输入(未示出),对一个或多个人类用户的输入做出响应。所述人机界面设备还可用于捕获某些媒体,气与人类有意识的输入不必直接相关,如音频(例如:语音、音乐、环境声音)、图像(例如:扫描图像、从静止影像相机获得的摄影图像)、视频(例如二维视频、包括立体视频的三维视频)。

[0182] 人机界面输入设备可包括以下中的一个或多个(仅绘出其中一个):键盘301、鼠标302、触控板303、触摸屏310以及相关的图形适配器350、数据手套、操纵杆305、麦克风306、扫描仪307、照相机308。

[0183] 计算机系统300还可以包括某些人机界面输出设备。这种人机界面输出设备可以通过例如触觉输出、声音、光和嗅觉/味觉来刺激一个或多个人类用户的感觉。这样的人机界面输出设备可包括触觉输出设备(例如通过触摸屏310、数据手套或操纵杆305的触觉反馈,但也可以有不用作输入设备的触觉反馈设备)、音频输出设备(例如,扬声器309、耳机(未示出))、视觉输出设备(例如,包括阴极射线管(CRT)屏幕、液晶显示(LCD)屏幕、等离子屏幕、有机发光二极管(OLED)屏的屏幕310,其中每一个都具有或没有触摸屏输入功能、每一个都具有或没有触觉反馈功能——其中一些可通过诸如立体画面输出的手段输出二维视觉输出或三维以上的输出;虚拟现实眼镜(未示出)、全息显示器和放烟箱(未示出))以及打印机(未示出)。

[0184] 计算机系统300还可以包括人可访问的存储设备及其相关介质,如包括具有CD/DVD的高密度只读/可重写式光盘(CD/DVD ROM/RW)320或类似介质321的光学介质、拇指驱动器322、可移动硬盘驱动器或固体状态驱动器323,诸如磁带和软盘(未示出)的传统磁介质,诸如安全软件保护器(未示出)等的基于ROM/ASIC/PLD的专用设备,等等。

[0185] 本领域技术人员还应当理解,结合所公开的主题使用的术语“计算机可读介质”不包括传输介质、载波或其它瞬时信号。

[0186] 计算机系统300还可以包括通往一个或多个通信网络(355)的一个或多个接口。例如,网络可以是无线的、有线的、光学的。网络还可为局域网、广域网、城域网、车载网络和工业网络、实时网络、延迟容忍网络等等。网络的示例还包括以太网、无线局域网、蜂窝网络(移动通信全球系统(GSM)、第三代(3G)、第四代(4G)、第五代(5G)、长期演进(LTE)等)等局

域网、电视有线或无线广域数字网络(包括有线电视、卫星电视、和地面广播电视)、车载和工业网络(包括CANBus)等等。某些网络通常需要外部网络接口适配器(354),用于连接到某些通用数据端口或外围总线(349)(例如,计算机系统(2200)的通用串行总线(USB)端口);其它系统通常通过连接到如下所述的系统总线集成到计算机系统300的核心(例如,以太网接口集成到PC计算机系统或蜂窝网络接口集成到智能电话计算机系统)。作为示例,网络355可以使用网络接口354连接到外围总线349。通过使用这些网络中的任何一个,计算机系统300可以与其它实体进行通信。所述通信可以是单向的,仅用于接收(例如,无线电视),单向的仅用于发送(例如CAN总线到某些CAN总线设备),或双向的,例如通过局域或广域数字网络到其它计算机系统。上述的每个网络和网络接口(354)可使用某些协议和协议栈。

[0187] 上述的人机界面设备、人可访问的存储设备以及网络接口可以连接到计算机系统300的核心340。

[0188] 核心340可包括一个或多个中央处理单元(CPU)341、图形处理单元(GPU)342、以现场可编程门阵列(FPGA)343形式的专用可编程处理单元、用于特定任务的硬件加速器344等。这些设备以及只读存储器(ROM)345、随机存取存储器(RAM)346、内部大容量存储器(例如内部非用户可存取硬盘驱动器、固态驱动器(SSD)等)347等可通过系统总线348进行连接。在某些计算机系统中,可以以一个或多个物理插头的形式访问系统总线348,以便可通过额外的中央处理单元、图形处理单元等进行扩展。外围装置可直接附接到核心的系统总线348,或通过外围总线349进行连接。外围总线的体系结构包括外围组件互联(PCI)、USB等。

[0189] CPU 341、GPU 341、FPGA 343和加速器344可以执行某些指令,这些指令组合起来可以构成上述计算机代码。该计算机代码可以存储在ROM 345或RAM 346中。过渡数据也可以存储在RAM 346中,而永久数据可以存储在例如内部大容量存储器347中。通过使用高速缓冲存储器可实现对任何存储器设备的快速存储和检索,高速缓冲存储器可与一个或多个CPU 341、GPU 342、大容量存储器347、ROM 345、RAM 346等紧密关联。

[0190] 所述计算机可读介质上可具有计算机代码,用于执行各种计算机实现的操作。介质和计算机代码可以是为本公开的目的而特别设计和构造的,也可以是计算机软件领域的技术人员所熟知和可用的介质和代码。

[0191] 作为实施例而非限制,具有体系结构的计算机系统300,特别是核心340,可以作为处理器(包括CPU、GPU、FPGA、加速器等)提供执行包含在一个或多个有形的计算机可读介质中的软件的功能。这种计算机可读介质可以是与上述的用户可访问的大容量存储器相关联的介质,以及具有非易失性的核心340的特定存储器,例如核心内部大容量存储器347或ROM 345。实现本公开的各种实施例的软件可以存储在这种设备中并且由核心340执行。根据特定需要,计算机可读介质可包括一个或一个以上存储设备或芯片。该软件可以使得核心340特别是其中的处理器(包括CPU、GPU、FPGA等)执行本文所述的特定过程或特定过程的特定部分,包括定义存储在RAM 346中的数据结构以及根据软件定义的过程来修改这种数据结构。另外或作为替代,计算机系统可以提供逻辑硬连线或以其它方式包含在电路(例如,加速器344)中的功能,该电路可以代替软件或与软件一起运行以执行本文所述的特定过程或特定过程的特定部分。在适当的情况下,对软件的引用可以包括逻辑,反之亦然。在适当的情况下,对计算机可读介质的引用可包括存储执行软件的电路(如集成电路(IC)),包含执

行逻辑的电路,或两者兼备。本公开包括任何合适的硬件和软件组合。

[0192] 虽然本公开已对多个示例性实施例进行了描述,但实施例的各种变更、排列和各种等同替换均属于本公开的范围内。因此应理解,本领域技术人员能够设计多种系统和方法,所述系统和方法虽然未在本文中明确示出或描述,但其体现了本公开的原则,因此属于本公开的精神和范围之内。

[0193] 非专利文献:

[0194] [1]IDF_10092019_视频编解码的高级语法控制_v2

[0195] 缩略词列表:

[0196] HLS:高级语法(High level syntax)

[0197] HEVC:高效视频编解码(High Efficiency Video Coding)

[0198] VVC:通用视频编解码(Versatile Video Coding)

[0199] CTU:编码树单元(Coding Tree Unit)

[0200] SPS:序列参数集(Sequence Parameter Set)

[0201] PPS:图片参数集(Picture Parameter Set)

[0202] APS:自适应参数集(Adaptive Parameter Set)

[0203] PH:图片头(Picture Header)

[0204] SH:切片头(Slice Header)

[0205] SAO:样本自适应偏移2(Sample Adaptive Offset2)

[0206] AU:访问单元(Access Unit)

[0207] NAL:网络抽象层(Network Abstraction Layer)

[0208] RBSP:原始字节序列载荷(Raw Byte Sequence Payload)

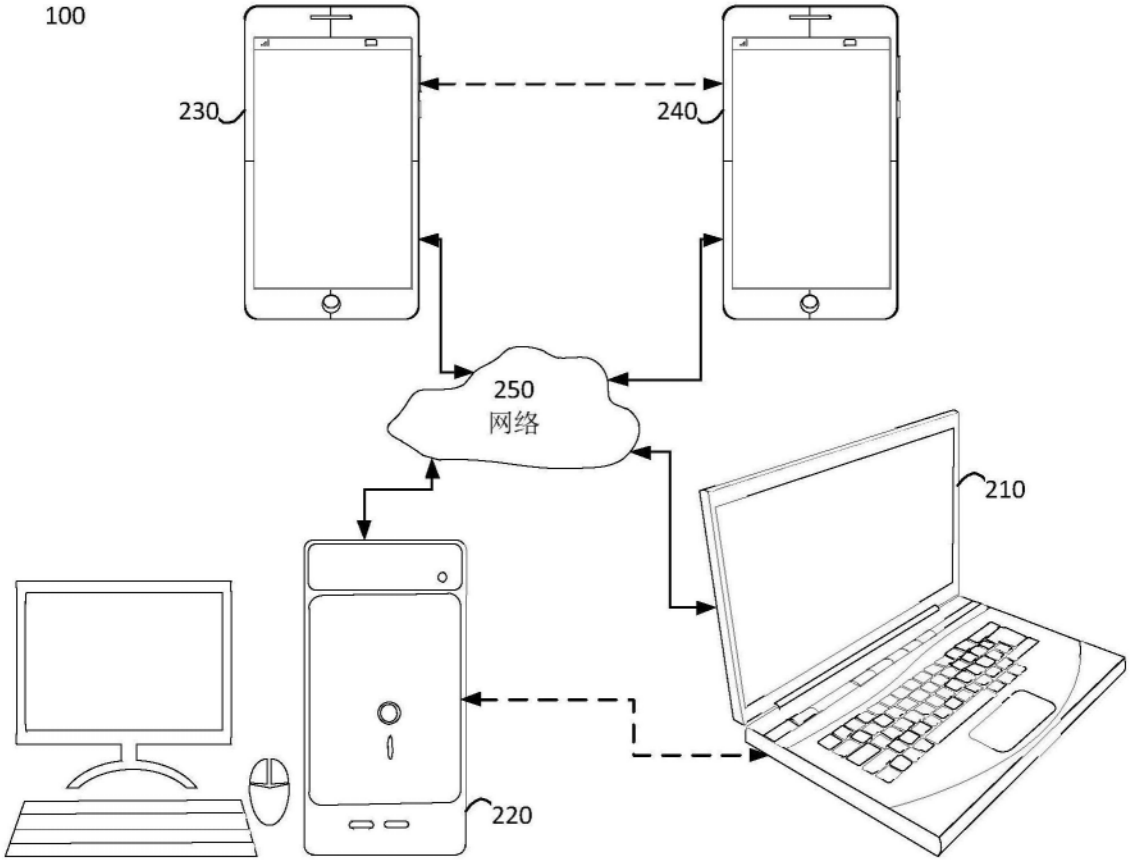


图1

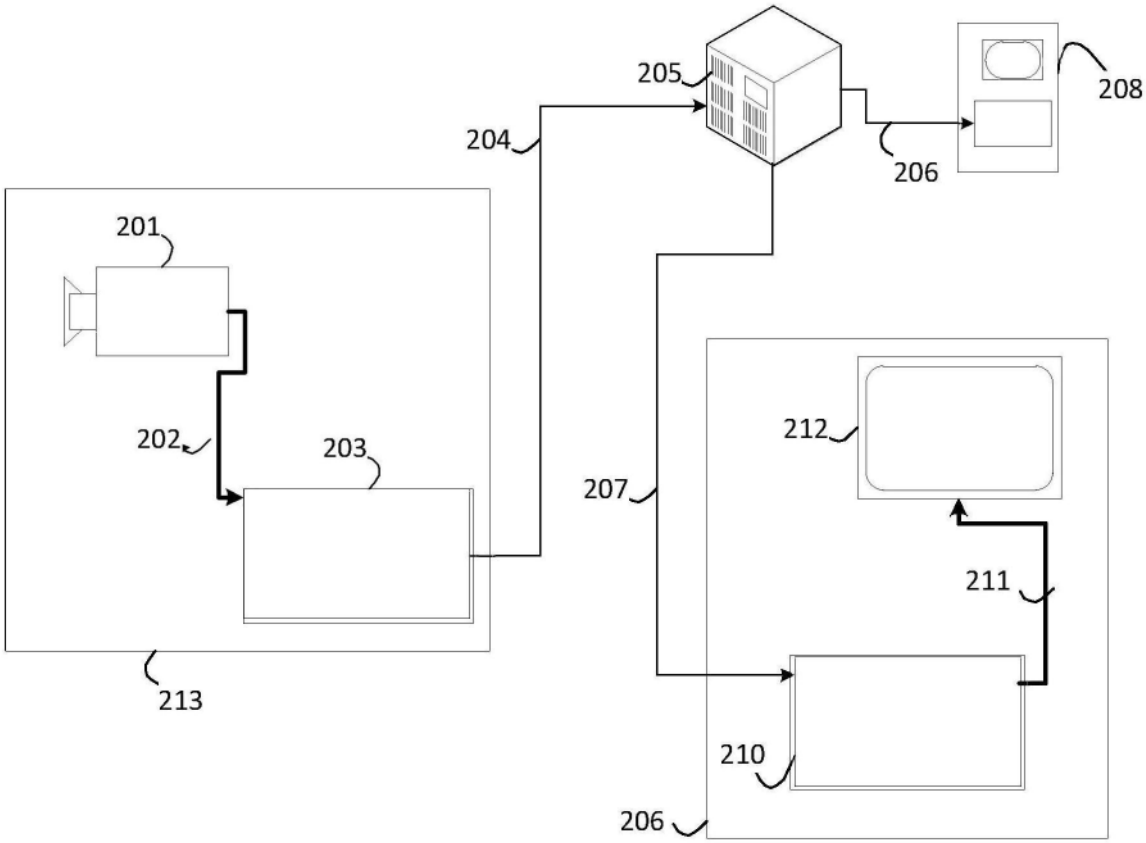


图2

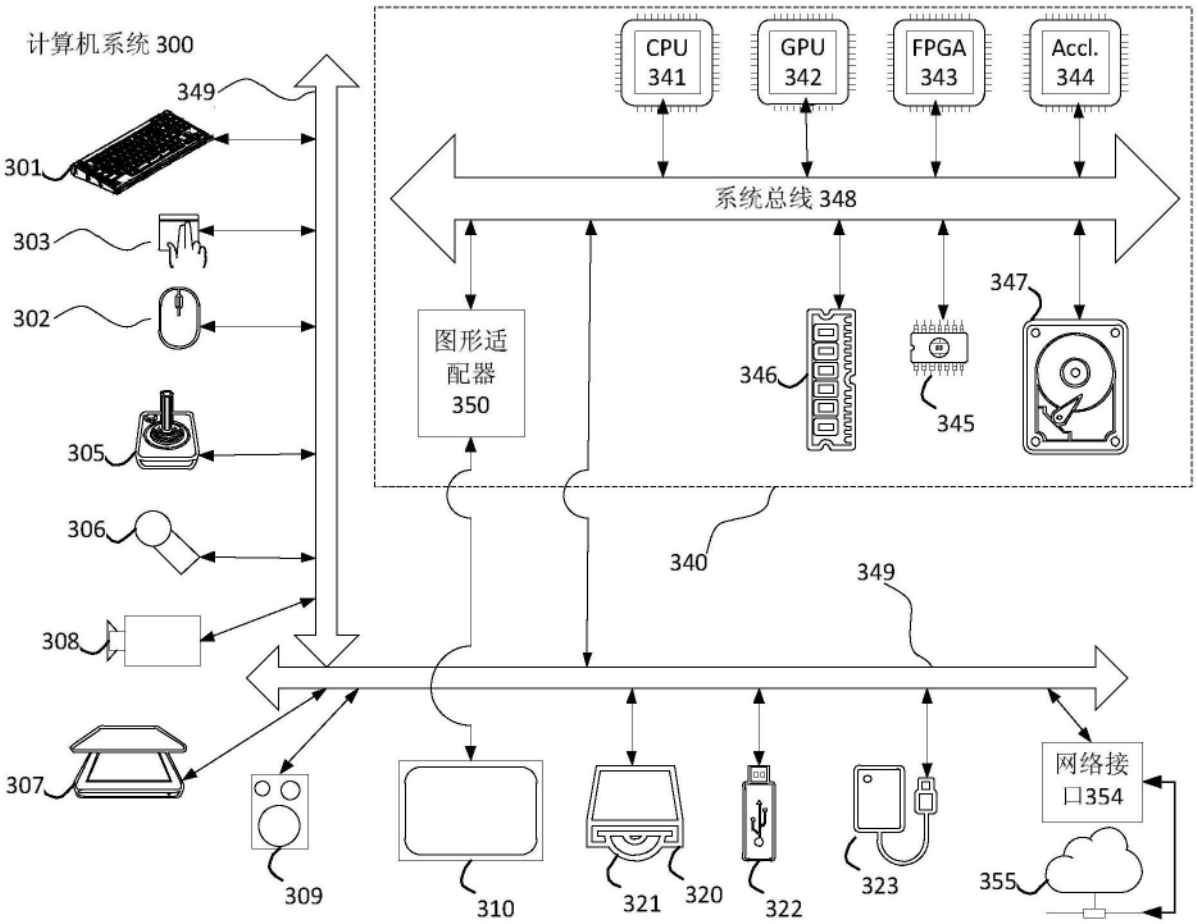

计算机系统 300
CPU 341
GPU 342
FPGA 343
Accl. 344
系统总线 348
图形适配器 350
346
345
347
340
349
301
303
302
305
306
308
307
309
310
321
320
322
323
网络接口354
355

图3