



# **CV180X & CV181X IVE API User Guide**

Version: 1.0

Release date: 2022-06-18

Copyright © 2020 CVITEK Co., Ltd. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of CVITEK Co., Ltd.

# Contents

<b>1</b>	<b>Disclaimer</b>	<b>2</b>
<b>2</b>	<b>Function Overview</b>	<b>3</b>
2.1	Objective . . . . .	3
2.2	Definitions and abbreviations . . . . .	3
<b>3</b>	<b>API reference</b>	<b>9</b>
3.1	Create Handle . . . . .	9
3.2	Destroy Handle . . . . .	9
3.3	DMA . . . . .	10
3.4	Filter . . . . .	11
3.5	Filter And CSC . . . . .	12
3.6	CSC . . . . .	13
3.7	Sobel . . . . .	14
3.8	NormGrad . . . . .	15
3.9	Canny Edge . . . . .	17
3.10	Canny Hysteresis Edge . . . . .	17
3.11	MagAndAng . . . . .	18
3.12	Dilate . . . . .	19
3.13	Erode . . . . .	20
3.14	Thresh . . . . .	21
3.15	And . . . . .	22
3.16	Sub . . . . .	23
3.17	Or . . . . .	24
3.18	Map . . . . .	25
3.19	OrdStatFilter . . . . .	26
3.20	Integral . . . . .	27
3.21	Histogram . . . . .	28
3.22	Add . . . . .	29
3.23	Xor . . . . .	30
3.24	Match BgModel . . . . .	31
3.25	Update BgModel . . . . .	32
3.26	Gradient of Foreground . . . . .	33
3.27	GMM . . . . .	34
3.28	GMM2 . . . . .	35
3.29	Bernsen . . . . .	36
3.30	NCC . . . . .	36
3.31	LBP . . . . .	37
3.32	SAD . . . . .	38
3.33	BufFlush . . . . .	40
3.34	BufRequest . . . . .	41

3.35	CreateMemInfo	41
3.36	CreatDataInfo	42
3.37	CreateImage	43
3.38	CreateImage with Cache	43
3.39	ResetImage	44
3.40	ReadImageArray	45
3.41	ReadMem	46
3.42	ReadMemArray	46
3.43	ReadData	47
3.44	ReadDataArray	48
3.45	ReadImage	48
3.46	ReadRawImage	49
3.47	WriteData	50
3.48	WriteMem	50
3.49	WriteImage	51
3.50	WriteRawImage	51
3.51	Reset Register	52
3.52	Dump Register	53
3.53	Split DiffFg of BgModel	53
3.54	Split ChgSta of BgModel	54
3.55	Query Tasks	54
3.56	Image2VideoFrameInfo	55
3.57	VideoFrameInfo2Image	56
3.58	FreeM	56
3.59	FreeI	57
3.60	FreeD	57
3.61	Thresh_S16	58
3.62	Thresh_U16	59
3.63	Resize	59
3.64	16BitTo8Bit	60
3.65	RGB YUV Erode to Dilate	61
3.66	STCandiCorner	62
3.67	Background Subtraction	63
<b>4</b>	<b>Data Type and Data Structure</b>	<b>65</b>
4.1	Define Data Types	67
4.2	Define Structure Type	67
4.2.1	IVE_IMAGE_TYPE_E_NUM	67
4.2.2	IVE_IMAGE_S	70
4.2.3	IVE_SRC_IMAGE_S	71
4.2.4	IVE_DST_IMAGE_S	71
4.2.5	IVE_DATA_S	72
4.2.6	IVE_SRC_DATA_S	73
4.2.7	IVE_DST_DATA_S	73
4.2.8	IVE_MEM_INFO_S	74
4.2.9	IVE_SRC_MEM_INFO_S	74
4.2.10	IVE_DST_MEM_INFO_S	75
4.2.11	IVE_8BIT_U	75
4.2.12	IVE_POINT_U16_S	76
4.2.13	IVE_POINT_S16_S	76
4.2.14	IVE_DMA_MODE_E	77

4.2.15	IVE_DMA_CTRL_S . . . . .	78
4.2.16	IVE_FILTER_CTRL_S . . . . .	79
4.2.17	IVE_CSC_MODE_E . . . . .	80
4.2.18	IVE_CSC_CTRL_S . . . . .	81
4.2.19	IVE_SOBEL_OUT_CTRL_E . . . . .	82
4.2.20	IVE_SOBEL_CTRL_S . . . . .	83
4.2.21	IVE_MAG_AND_ANG_OUT_CTRL_E . . . . .	83
4.2.22	IVE_MAG_AND_ANG_CTRL_S . . . . .	84
4.2.23	IVE_DILATE_CTRL_S . . . . .	85
4.2.24	IVE_ERODE_CTRL_S . . . . .	85
4.2.25	IVE_THRESH_MODE_E . . . . .	86
4.2.26	IVE_THRESH_CTRL_S . . . . .	87
4.2.27	IVE_SUB_MODE_E . . . . .	89
4.2.28	IVE_SUB_CTRL_S . . . . .	89
4.2.29	IVE_INTEG_OUT_CTRL_E . . . . .	90
4.2.30	IVE_INTEG_CTRL_S . . . . .	91
4.2.31	IVE_THRESH_S16_MODE_E . . . . .	91
4.2.32	IVE_THRESH_S16_CTRL_S . . . . .	92
4.2.33	IVE_THRESH_U16_MODE_E . . . . .	93
4.2.34	IVE_THRESH_U16_CTRL_S . . . . .	94
4.2.35	IVE_16BIT_TO_8BIT_MODE_E . . . . .	95
4.2.36	IVE_16BIT_TO_8BIT_CTRL_S . . . . .	96
4.2.37	IVE_ORD_STAT_FILTER_MODE_E . . . . .	97
4.2.38	IVE_ORD_STAT_FILTER_CTRL_S . . . . .	97
4.2.39	IVE_MAP_MODE_E . . . . .	98
4.2.40	IVE_ADD_CTRL_S . . . . .	99
4.2.41	IVE_NCC_DST_MEM_S . . . . .	99
4.2.42	IVE_GMM_CTRL_S . . . . .	100
4.2.43	IVE_LBP_CMP_MODE_E . . . . .	103
4.2.44	IVE_LBP_CTRL_S . . . . .	103
4.2.45	IVE_NORM_GRAD_OUT_CTRL_E . . . . .	104
4.2.46	IVE_NORM_GRAD_CTRL_S . . . . .	105
4.2.47	IVE_SAD_MODE_E . . . . .	106
4.2.48	IVE_SAD_OUT_CTRL_E . . . . .	106
4.2.49	IVE_SAD_CTRL_S . . . . .	107
4.2.50	IVE_HOG_CTRL_S . . . . .	109
4.2.51	IVE_GRAD_FG_CTRL_S . . . . .	109
4.2.52	IVE_GRAD_FG_MODE_E . . . . .	110
4.2.53	IVE_16BIT_TO_8BIT_MODE_E . . . . .	111
4.2.54	IVE_16BIT_TO_8BIT_CTRL_S . . . . .	112
4.2.55	IVE_IVE_TYPE_E . . . . .	113
4.2.56	IVE_IVE_CTRL_S . . . . .	114
4.2.57	IVE_BLOCK_CTRL_S . . . . .	114
<b>5</b>	<b>Tips Description</b>	<b>116</b>
5.1	The additional Buffer . . . . .	116
<b>6</b>	<b>FAQ</b>	<b>117</b>
6.1	The Use of The Cache . . . . .	117
6.2	The config of the blnstant parameter . . . . .	117

**Revision History**

Revision	Date	Description
1.0	2022/06/18	first edition

# 1 Disclaimer

---



## Terms and Conditions

The document and all information contained herein remain the CVITEK Co., Ltd' s ( "CVITEK" ) confidential information, and should not disclose to any third party or use it in any way without CVITEK' s prior written consent. User shall be liable for any damage and loss caused by unauthority use and disclosure.

CVITEK reserves the right to make changes to information contained in this document at any time and without notice.

All information contained herein is provided in "AS IS" basis, without warranties of any kind, expressed or implied, including without limitation mercantability, non-infringement and fitness for a particular purpose. In no event shall CVITEK be liable for any third party' s software provided herein, User shall only seek remedy against such third party. CVITEK especially claims that CVITEK shall have no liable for CVITEK' s work result based on Customer' s specification or published shandard.

## Contact Us

**Address** Building 1, Yard 9, FengHao East Road, Haidian District, Beijing, 100094, China

Building T10, UpperCoast Park, Huizhanwan, Zhancheng Community, Fuhai Street, Baoan District, Shenzhen, 518100, China

**Phone** +86-10-57590723 +86-10-57590724

**Website** <https://www.sophgo.com/>

**Forum** <https://developer.sophgo.com/forum/index.html>

# 2 Function Overview

---

## 2.1 Objective

Intelligent Video Engine (IVE) is a module that uses hardware to accelerate computer vision algorithms.

Users can develop intelligent analysis solutions using IVE to speed up analysis calculations and reduce processor usage.

Currently, the operators provided by IVE support the development of intelligent analysis solutions for images or videos.

(Please note that IVE is supported on cv181x, but not on cv180x.)

## 2.2 Definitions and abbreviations

- handle

When users call operators to create tasks, the system will assign a handle to each task to identify the execution status of different tasks.

- Timely return result flag (bInstant)

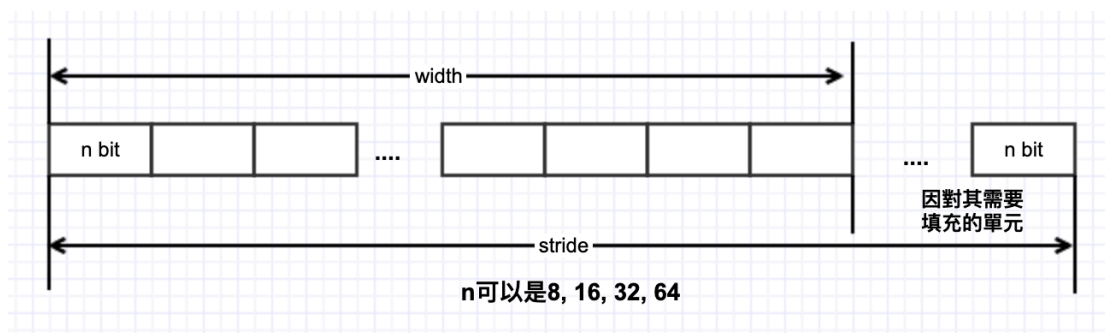
“True “indicates Busy waiting mode, and ” False “indicates Interrupt mode.

- stride

The amount corresponding to the width of the image or two-dimensional data is shown in Figure 1-1

- IVE\_IMAGE\_S image data stride, which indicates the number of units in a line of image calculated by “pixel” , and the bit width of “pixel” can be 8bit, 16bit, etc.
- IVE\_DATA\_S two-dimensional data stride, which represents the number of bytes in a row of two-dimensional data, is the case of  $n = 8$  in Figure 1-1.

*Figure 1-1 schematic diagram of stride*



- alignment

In order to quickly access the first memory address or cross row access data, hardware requires that the memory address or memory stride must be a multiple of the alignment factor.

- Data memory first address alignment

Currently, IVE operators require 16 pixels alignment for their input and output.

- stride alignment

The stride of two-dimensional generalized image, two-dimensional single component data and one-dimensional array data must meet the 16 pixels alignment

*input and output data types*

Types	Image Description	Memory Address	Stride
IVE_IMAGE_TYPE	8bit single channel image without sign Figure 1-2	only u64PhyAddr[0]、u64VirAddr[0] in IVE_IMAGE_S is used	only u32Stride[0] is used
IVE_IMAGE_TYPE	8bit single channel image with sign Figure 1-2	only u64PhyAddr[0]、u64VirAddr[0] in IVE_IMAGE_S is used	only u32Stride[0] is used
IVE_IMAGE_TYPE	YUV420P Planar Data format image, figure	u64PhyAddr[0]、u64VirAddr[0](brightness Y), u64PhyAddr[1]、u64VirAddr[1](chroma U, V) in IVE_IMAGE_S is used in memory address	Stride uses u32Stride[0](brightness stride)、u32Stride[1](chroma U, V stride)

continues on next page



Table 2.1 – continued from previous page

Types	Image Description	Memory Address	Stride
IVE_IMAGE_TYPE_YUV422P2	Planar Data format image, figure	u64PhyAddr[0], u64VirAddr[0](brightness Y), u64PhyAddr[1], u64VirAddr[1](chroma U, V) in IVE_IMAGE_S are used in memory address	Stride uses u32Stride[0](brightness stride), u32Stride[1](chroma U, V stride)
IVE_IMAGE_TYPE_YUV420P0	Planar Data format image, figure 1-3	u64PhyAddr[0], u64VirAddr[0](brightness Y), u64PhyAddr[1], u64VirAddr[1](chroma U) and u64PhyAddr[2], u64VirAddr[2](chroma V) in IVE_IMAGE_S are used in memory address	Stride uses u32Stride[0](brightness stride), u32Stride[1](chroma U stride) and u32Stride[2](chroma V stride)
IVE_IMAGE_TYPE_YUV422P2	Planar Data format image, figure 1-4	u64PhyAddr[0], u64VirAddr[0](brightness Y), u64PhyAddr[1], u64VirAddr[1](chroma U) and u64PhyAddr[2], u64VirAddr[2](chroma V) in IVE_IMAGE_S are used in memory address	Stride uses u32Stride[0](brightness stride), u32Stride[1](chroma U stride) and u32Stride[2](chroma V stride)
IVE_IMAGE_TYPE_U8G2_UNPACKED	Unsigned Integer with 2 channels stored in packed format, Figure	Only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used in memory address	Stride only used u32Stride[0]
IVE_IMAGE_TYPE_U8G2_PLANAR	Unsigned Integer with 2 channels stored in planar format, Figure	Only u64PhyAddr[0], u64VirAddr[0], u64PhyAddr[1], u64VirAddr[1] are used in memory address	Stride only used u32Stride[0], u32Stride[1]
IVE_IMAGE_TYPE_I16G1	signed integer with single channel	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used

continues on next page

Table 2.1 – continued from previous page

Types	Image Description	Memory Address	Stride
IVE_IMAGE_TYPE_U16C1	16-bit unsigned integer with single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_U8C3_PACKED	8-bit unsigned integer with 3 channels stored in packed format, Figure 1-5	Only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used in memory address	Stride only u32Stride[0] used
IVE_IMAGE_TYPE_U8C3_PLANAR	8-bit unsigned integer with 3 channels stored in planar format, Figure 1-6	Only u64PhyAddr[0], u64VirAddr[0], u64PhyAddr[1], u64VirAddr[1], u64PhyAddr[2], u64VirAddr[2] in IVE_IMAGE_S are used in memory address	Stride only u32Stride[0], u32Stride[1], u32Stride[2] used
IVE_IMAGE_TYPE_S16C1	16-bit signed integer with single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_U16C1	16-bit unsigned integer with single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_S16C1	16-bit signed integer with single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_U16C1	16-bit unsigned integer with single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_F16C1	16-bit brain floating point single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used
IVE_IMAGE_TYPE_F32C1	32-bit floating point single channel, figure 1-2	only u64PhyAddr[0], u64VirAddr[0] in IVE_IMAGE_S are used	only u32Stride[0] used

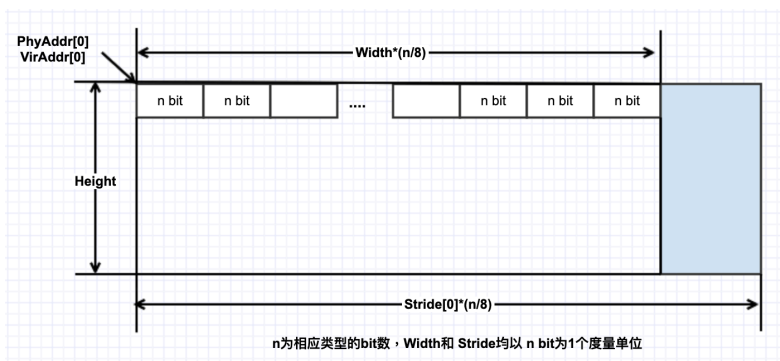


Fig. 2.1: Single-channel Image

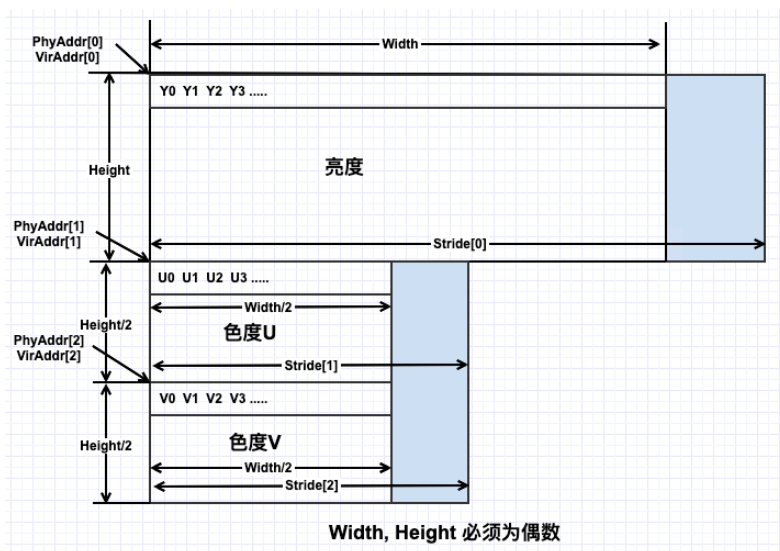


Fig. 2.2: IVE\_IMAGE\_TYPE\_YUV420P type of IVE\_IMAGE\_S image

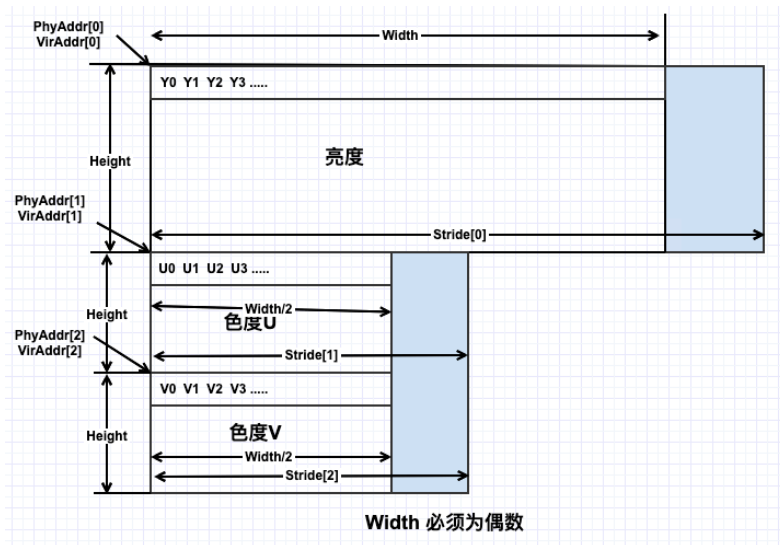


Fig. 2.3: IVE\_IMAGE\_TYPE\_YUV422P type of IVE\_IMAGE\_S image

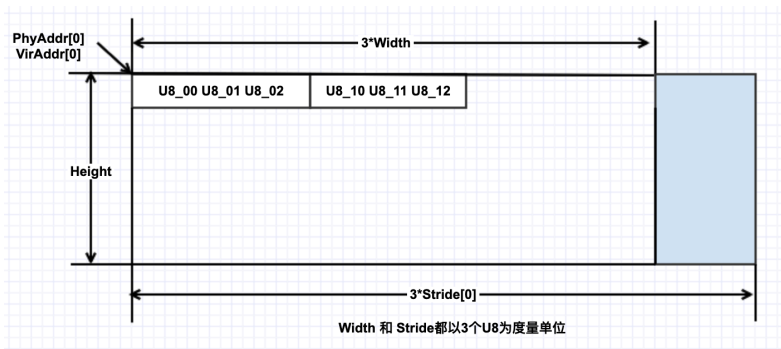


Fig. 2.4: `IVE_IMAGE_TYPE_U8C3_PACKAGE` type of `IVE_IMAGE_S` image

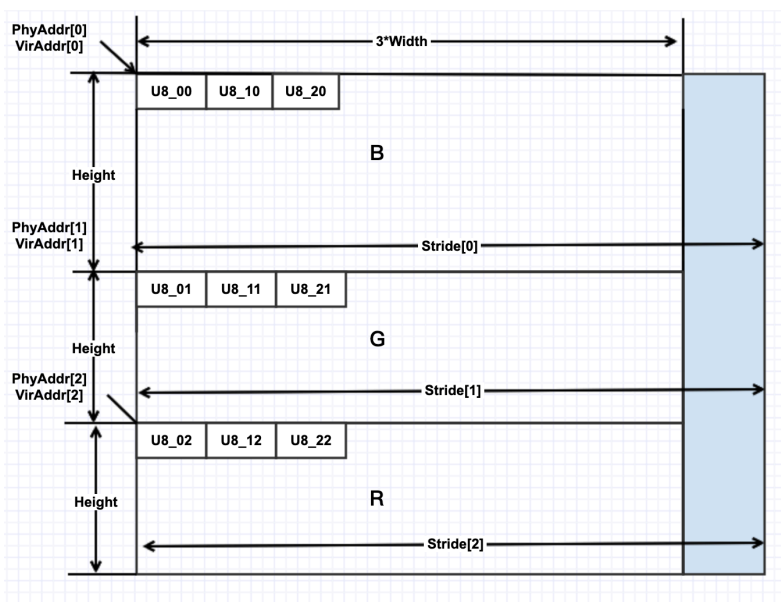


Fig. 2.5: `IVE_IMAGE_TYPE_U8C3_PLANAR` type of `IVE_IMAGE_SRC` image

# 3 API reference

## 3.1 Create Handle

**【Description】**

Create IVE handle.

**【Syntax】**

```
IVE_HANDLE CVI_IVE_CreateHandle();
```

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.2 Destroy Handle

**【Description】**

Free IVE handel.

**【Syntax】**

```
CVI_S32 CVI_IVE_DestroyHandle(IVE_HANDLE pIveHandle);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer, cannot be null	Input

**【Requirement】**

- Header files:cvi\_comm\_ive.h cvi\_ive.h

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

## 3.3 DMA

### 【Description】

Create direct memory access task, support fast copy, interval copy and memory filling:  
it can realize the fast copy of data from one memory to another,  
or regularly copy some data from one memory to another, or fill one memory.

### 【Syntax】

```
CVI_S32 CVI_IVE_DMA(IVE_HANDLE pIveHandle, IVE_DST_DATA_S *pstSrc, IVE_DST_DATA_S *pstDst, IVE_DMA_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output data pointer. Cannot be empty in copy mode.	Output
pstCtrl	DMA control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time. True indicates busy waiting mode, False indicates interrupt mode	Input

Parameter	Support Type	Address Alignment	Resolution
pstSrc	IVE_DATA_S	1 byte	32x1~1920x1080
pstDst	IVE_DST_DATA_S	1 byte	Same as pstSrc for direct copying. Smaller than pstSrc for interval copying.

### 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.4 Filter

### 【Description】

Create a 5x5 template filtering task, and configure different template coefficients to achieve different filtering task.

### 【Syntax】

```
CVI_S32 CVI_IVE_Filter(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_FILTER_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output data pointer. The width and height are the same as pstSrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1、YUV420SP、YUV422SP	16 byte	64x64~1920x1024
pstDst	Same as pstSrc	16 byte	Same as pstSrc

### 【Return Value】

Return value	Description
0	Success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

### 【Note】

## 3.5 Filter And CSC

### 【Description】

Create a 5x5 template filtering and YUV2RGB color space conversion task.

Different filters can be achieved by configuring different template coefficients.

### 【Syntax】

```
CVI_S32 CVI_IVE_FilterAndCSC(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_FILTER_AND_CSC_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output data pointer. The width and height are the same as pstSrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	YUV420SP、YUV422SP	16 byte	64x64~1920x1024
pstDst	U8C3_PLANAR or U8C3_PACKAGE	16 byte	Same as pstSrc

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

### 【Note】



## 3.6 CSC

### 【Description】

Create the color space conversion task.

### 【Syntax】

```
CVI_S32 CVI_IVE_CSC(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_
↪IMAGE_S *pstDst, IVE_FILTER_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output data pointer. The width and height are same as pstSrc.	Output
pstCscCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	YUV420SP、 YUV422SP、 U8C3_PLANAR、 U8C3_PACKAGE	6 byte	4x64~1920x1024
pstDst	U8C3_PLANAR、 U8C3_PACKAGE、 YUV420SP、 YUV422SP	16 byte	Same as pstSrc

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cv\_i\_comm\_ive.h cv\_i\_ive.h

### 【Note】

## 3.7 Sobel

### 【Description】

Create a 5x5 template Sobel like gradient calculation task.

### 【Syntax】

```
CVI_S32 CVI_IVE_Sobel(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDstH, IVE_DST_IMAGE_S *pstDstV, IVE_SOBEL_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDstH	H pointer of gradient component image obtained by template direct filtering. Root pstSobelCtrl→enOutCtrl. If output is required, it cannot be empty. The width and height are the same as pstSrc.	Output
pstDstV	The V pointer of gradient component image obtained by template direct filtering. Root pstSobelCtrl→enOutCtrl. If output is required, it cannot be empty. The width and height are the same as pstsrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	8x8~1920x1024
pstDstH	S16C1	16 byte	Same as pstSrc
pstDstV	S16C1	16 byte	Same as pstSrc

### 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `file:cvl_comm_ive.h` `cvl_ive.h`

**【Note】**

## 3.8 NormGrad

**【Description】**

Create a task for normalized gradient calculation.

All gradients will be normalized to S8 format.

**【Syntax】**

```
CVI_S32 CVI_IVE_NormGrad(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_
↪ DST_IMAGE_S *pstDstH, IVE_DST_IMAGE_S *pstDstV, IVE_DST_IMAGE_S *pstDstHV,
↪ IVE_NORM_GRAD_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDstH	H pointer of gradient component image obtained by template direct filtering. Root pstNormGradCtrl→enOutCtrl. If output is required, it cannot be empty. The width and height are the same as pstSrc.	Output
pstDstV	The V pointer of gradient component image obtained by template direct filtering. Root pstNormGradCtrl→enOutCtrl. If output is required, it cannot be empty. The width and height are the same as pstsrc.	Output
pstDstHV	HV pointer of gradient component image obtained by template direct filtering. Root pstNormGradCtrl→enOutCtrl. If output is required, it cannot be empty. The width and height are the same as pstSrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	
pstDstH	S8C1	16 byte	
pstDstV	S8C1	16 byte	
pstDstHV	S8C2_PACKAGE	16 byte	

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

【Note】

## 3.9 Canny Edge

【Description】

Link the edges of the Canny image.

【Syntax】

```
CVI_S32 CVI_IVE_CannyEdge(IVE_IMAGE_S *pstEdge, IVE_MEM_INFO_S *pstStack);
```

【Parameter】

Parameter	Description	Input/Output
pstEdge	Input an Edge Flag image, output a binary boundary image.	Input/output
pstStack	Coordinates of strong edges.	Input/output

Parameter	Support Image Type	Address Alignment	Resolution
pstEdge	U8C1	16 byte	
pstStack	•	16 byte	

【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

【Requirement】

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

【Note】

## 3.10 Canny Hysteresis Edge

【Description】

Create a Canny Edge task, calculating the grayscale image' s Gradient, Gradient Magnitude, Hysteresis threshold and Non-Maximum Suppression.

【Syntax】

```
CVI_S32 CVI_IVE_CannyHysEdge(IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstSrc, IVE_
↪DST_IMAGE_S *pstEdge, IVE_MEM_INFO_S *pstStack, IVE_CANNY_HYS_EDGE_CTRL_S
↪*pstCtrl, CVI_BOOL bInstant);
```

#### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstEdge	Strong/Weak Edge Flag image	Output
pstStack	Coordinates of strong edges.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	
pstEdge	U8C1	16 byte	
pstStack	•	16 byte	

#### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

#### 【Requirement】

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

#### 【Note】

## 3.11 MagAndAng

#### 【Description】

Create 5x5 template gradient amplitude and phase angle calculation task.

#### 【Syntax】

```
CVI_S32 CVI_IVE_MagAndAng(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_
↪DST_IMAGE_S *pstDstMag, IVE_DST_IMAGE_S *pstDstAng, IVE_MAG_AND_ANG_CTRL_S
↪*pstCtrl, CVI_BOOL bInstant);
```

## 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDstMag	Output amplitude image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstDstAng	Output phase angle image pointer. According to pstMagAndAngCtrl→enOutCtrl, output cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDstMag	U16C1	16 byte	Same as pstSrc
pstDstAng	U8C1	16 byte	Same as pstSrc

## 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

## 【Requirement】

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.12 Dilate

## 【Description】

Create the binary image 5x5 template expansion task.

## 【Syntax】

```
CVI_S32 CVI_IVE_Dilate(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_DILATE_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

## 【Parameter】

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output amplitude image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDst	U8C1	16 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.13 Erode

**【Description】**

Create the binary image 5x5 template corrosion task.

**【Syntax】**

```
CVI_S32 CVI_IVE_Erode(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_ERODE_CTRL_S *pstErodeCtrl, CVI_BOOL bInstant);
```

**【Parameter】**



Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output amplitude image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstErodeCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	Binary image of U8C1	16 byte	6 4x64~1920x1024
pstDst	Binary image of U8C1	16 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.14 Thresh

**【Description】**

Create the grayscale image thresholding task.

**【Syntax】**

```
CVI_S32 CVI_IVE_Thresh(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_MAG_AND_ANG_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	Output amplitude image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDstMag	U16C1	16 byte	Same as pstSrc
pstDstAng	U8C1	16 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvl\_comm\_ive.h cvl\_ive.h

## 3.15 And

**【Description】**

Create a task to perform bitwise AND operation between two binary images.

**【Syntax】**

```
CVI_S32 CVI_IVE_And(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, IVE_AND_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty.	Input
pstSrc2	Source image 2 pointer. Cannot be empty.	Output
pstDst	Output image pointer. Cannot be empty. The height and width are the same as pstSrc1.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	Binary image of U16C1	1 byte	64x64~1920x1024
pstSrc2	U16C1	1 byte	Same as pstSrc
pstDst	U8C1	1 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.16 Sub

**【Description】**

Create a task to perform subtraction operation between two grayscale images.

**【Syntax】**

```
CVI_S32 CVI_IVE_Sub(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, IVE_SUB_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty.	Input
pstSrc2	Source image 2 pointer. Cannot be empty.	Output
pstDst	Output image pointer. Cannot be empty. The height and width are the same as pstSrc1.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	Binary image of U16C1	1 byte	64x64~1920x1024
pstSrc2	U16C1	1 byte	Same as pstSrc
pstDst	U8C1	1 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.17 Or

**【Description】**

Create a task to perform bitwise OR operation between two binary images.

**【Syntax】**

```
CVI_S32 CVI_IVE_Or(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, IVE_OR_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty.	Input
pstSrc2	Source image 2 pointer. Cannot be empty.	Output
pstDst	Output image pointer. Cannot be empty. The height and width are the same as pstSrc1.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	Binary image of U16C1	1 byte	64x64~1920x1024
pstSrc2	U16C1	1 byte	Same as pstSrc
pstDst	U8C1	1 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cv_i_comm_ive.h` `cv_i_ive.h`

## 3.18 Map

**【Description】**

Map an image onto another image through a lookup table.

**【Syntax】**

```
CVI_S32 CVI_IVE_Map(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_SRC_MEM_
↪INFO_S *pstMap, IVE_DST_IMAGE_S *pstDst, IVE_MAP_CTRL_S *pstCtrl, CVI_BOOL_
↪bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	The handle of the task.	Input
pstSrc	The pointer of the entered image. Cannot be empty.	Input
pstMap	the pointer of the entered mapping table. Cannot be empty.	Input
pstDst	The pointer of the outputted image. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cv\_i\_comm\_ive.h cv\_i\_ive.h

## 3.19 OrdStatFilter

**【Description】**

Find the maximum and minimum in the picture with 3x3 kernel.

**【Syntax】**

```
CVI_S32 CVI_IVE_OrdStatFilter(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc,
↪ IVE_DST_IMAGE_S *pstDst, IVE_ORD_STAT_FILTER_CTRL_S *pstCtrl, CVI_BOOL
↪ bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	The handle of the task.	Input
pstSrc	The pointer of the entered image. Cannot be empty.	Input
pstDst	The pointer of the outputted image. Cannot be empty. The height and width are the same as pstSrc	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files:cvi\_comm\_ive.h cvi\_ive.h

## 3.20 Integral

**【Description】**

Create a task to compute the integral image of a grayscale image.

**【Syntax】**

```
CVI_S32 CVI_IVE_Integ(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_MEM_INFO_S *pstDst, IVE_INTEG_CTRL_S *pstIntegCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	The pointer of the outputted amplitude image. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDst	U32C1, U64C1	16 byte	Same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.21 Histogram

**【Description】**

Create a histogram statistics task for a grayscale image.

**【Syntax】**

```
CVI_S32 CVI_IVE_Hist(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_MEM_INFO_S *pstDst, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	Source data pointer. Cannot be empty.	Input
pstDst	The pointer of the outputted amplitude image. Cannot be empty. The height and width are the same as pstSrc.	Output
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDst	•	16 byte	•

**【Return Value】**



Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `cvl_comm_ive.h` `cvl_ive.h`

## 3.22 Add

**【Description】**

Create a weighted sum calculation task for two grayscale images.

**【Syntax】**

```
CVI_S32 CVI_IVE_Add(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, , IVE_SRC_
→IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, IVE_ADD_CTRL_S *pstCtrl, CVI_BOOL_
→bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc1	Source data 1 pointer. Cannot be empty.	Input
pstSrc2	Source data 2 pointer. Cannot be empty.	Input
pstDst	The pointer of the outputted amplitude image. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	16 byte	64x64~1920x1024
pstDst	•	16 byte	•

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.23 Xor

**【Description】**

Create an XOR computation task for two binary images.

**【Syntax】**

```
CVI_S32 CVI_IVE_Xor(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty	Input
pstSrc2	Source image 2 pointer. Cannot be empty. The same height and width as pstSrc1	
pstDst	The pointer of the outputted image. Cannot be empty. The height and width are the same as pstSrc1.	Output
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	U8C1	1byte	64x64~1920x1024
pstSrc2	U8C1	1byte	Same as pstSrc
pstDst	•	1byte	-same as pstSrc

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.24 Match BgModel

### 【Description】

Input the current image and the model to obtain foreground data.

### 【Syntax】

```
CVI_S32 CVI_IVE_MatchBgModel(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstCurImg,
↪IVE_DATA_S *pstBgModel, IVE_IMAGE_S *pstFgFlag, IVE_DST_IMAGE_S *pstDiffFg,
↪IVE_DST_MEM_INFO_S *pstStatData, IVE_MATCH_BG_MODEL_CTRL_S *pstCtrl, CVI_BOOL
↪bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstCurImg	The current image.	Input
pstBgModel	The model.	Input/output
pstFgFlag	The foreground state image.	Input/output
pstDiffFg	The foreground image.	Output
pstStatData	The foreground state.	Output
pstCtrl	The control structure.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstCurImg	U8C1	1byte	
pstBgModel	•	1byte	
pstFgFlag	U8C1	1byte	
pstDiffFg	S8C1	1byte	
pstStatData	•	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.25 Update BgModel

### 【Description】

Update the background model.

### 【Syntax】

```
CVI_S32 CVI_IVE_UpdateBgModel(IVE_HANDLE pIveHandle, IVE_DATA_S *pstBgModel, IVE_IMAGE_S *pstFgFlag, IVE_DST_IMAGE_S *pstBgImg, IVE_DST_IMAGE_S *pstChaSta, IVE_DST_MEM_INFO_S *pstStatData, IVE_MATCH_BG_MODEL_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstBgModel	The model	Input/output
pstFgFlag	The foreground state image.	Input/output
pstBgImg	The background image.	Output
pstChaSta	Update the foreground state image.	Output
pstStatData	the background state	Output
pstCtrl	the control structure	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstBgModel	•	1byte	
pstFgFlag	U8C1	1byte	
pstBgImg	U8C1	1byte	
pstChaSta	S8C1	1byte	
pstStatData	•	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.26 Gradient of Foreground

### 【Description】

Calculate the foreground gradient image based on the background gradient image and the current image.

### 【Syntax】

```
CVI_S32 CVI_IVE_GradFg(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstBgDiffFg, IVE_
↪SRC_IMAGE_S *pstCurGrad, IVE_SRC_IMAGE_S *pstBgGrad, IVE_DST_IMAGE_S ↪
↪*pstGradFg, IVE_GRAD_FG_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstBgDiffFg	the foreground image	Input
pstCurGrad	the current gradient image	Input
pstBgGrad	the background gradient image	Input
pstGradFg	the foreground gradient image	Output
pstCtrl	the control structure	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstBgDiffFg	S8C1	1byte	
pstCurGrad	S8 C2_PACKAGE	1byte	
pstBgGrad	S8 C2_PACKAGE	1byte	
pstGradFg	S8C1	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.27 GMM

### 【Description】

Create a task to establish a GMM background model.

### 【Syntax】

```
CVI_S32 CVI_IVE_GMM(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_
↪IMAGE_S *pstFg, IVE_DST_IMAGE_S *pstBg, IVE_MEM_INFO_S *pstModel, IVE_GMM_CTRL_
↪S *pstCtrl, CVI_BOOL
bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	the inputted image	Input
pstFg	the foreground image	Output
pstBg	the background image	Output
pstModel	the model data	Input/output
pstCtrl	the control structure	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstModel	•	1byte	
pstSrc	U8C1 or U8C3_PACKAGE	1byte	
pstFg	U8C1 binary image	1byte	
pstBg	U8C1 or U8C3_PACKAGE	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.28 GMM2

### 【Description】

Create a task to establish a GMM background model.

### 【Syntax】

```
CVI_S32 CVI_IVE_GMM2(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_SRC_
↪IMAGE_S *pstFactor, IVE_DST_IMAGE_S *pstFg, IVE_DST_IMAGE_S *pstBg, IVE_DST_
↪IMAGE_S *pstMatchModelInfo, IVE_MEM_INFO_S *pstModel, IVE_GMM_CTRL_S_
↪*pstCtrl, CVI_BOOL
bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	the inputted image	Input
pstFactor	the model updated coefficient	Input
pstFg	the foreground image	Output
pstBg	the background image	Output
pstMatchModelInfo	the model matching coefficient	Output
pstModel	the model data	Input/output
pstCtrl	the control structure	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstModel	•	1byte	
pstFactor	U16C1	1byte	
pstSrc	U8C1 or U8C3_PACKAGE	1byte	
pstFg	U8C1 binary image	1byte	
pstBg	U8C1 or U8C3_PACKAGE	1byte	
pstMatchModelInfo	U8C1	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.29 Bernsen

### 【Description】

Create a task to establish the Bernsen binarization algorithm.

### 【Syntax】

```
CVI_S32 CVI_IVE_Bernsen(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_
↪IMAGE_S *pstDst, IVE_BERSEN_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	the inputted image	Input
pstDst	the result image	Output
pstCtrl	the control structure	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc	U8C1	1byte	
pstDst	U8C1 binary image	1byte	

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.30 NCC

### 【Description】

Create a task to calculate the normalized cross-correlation coefficient for two gray-scale images with the same resolution.

### 【Syntax】

```
CVI_S32 CVI_IVE_NCC(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_
↪IMAGE_S *pstSrc2, IVE_DST_MEM_INFO_S *pstDst, CVI_BOOL bInstant);
```



**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty.	Input
pstSrc2	Source image 2 pointer. Cannot be empty.	Input
pstDst	Output data pointer. Cannot be empty. Memory should be configured at least: sizeof (IVE_NCC_DST_MEM_S)。	Output
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	U8C1	1byte	64x64~1920x1024
pstSrc2	U8C1	1byte	Same as pstSrc
pstDst	•	16 byte	•

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.31 LBP

**【Description】**

Create an LBP computation task.

**【Syntax】**

```
CVI_S32 CVI_IVE_LBP(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S * pstDst, IVE_LBP_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc	Source image pointer. Cannot be empty.	Input
pstDst	Output data pointer. Cannot be empty. Memory should be configured at least: sizeof (IVE_NCC_DST_MEM_S)。	Output
pstCtrl	Control parameter pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	U8C1	1byte	64x64~1920x1024
pstSrc2	U8C1	1byte	Same as pstSrc
pstDst	•	16 byte	•

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.32 SAD

**【Description】**

Calculate the SAD between two images.

**【Syntax】**

```
CVI_S32 CVI_IVE_SAD(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1, IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstSad, IVE_DST_IMAGE_S *pstThr, IVE_SAD_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	Handle pointer. Cannot be empty.	Input
pstSrc1	Source image 1 pointer. Cannot be empty.	Input
pstSrc2	Source 2 image pointer. Cannot be empty. The height and width are the same as pstSrc1	Input
pstSad	Output SAD image pointer. According to <code>pstSadCtrl→enOutCtrl</code> , if output is required, it cannot be empty. According to <code>pstSadCtrl→enMode</code> , corresponding to 4x4, 8x8 and 16x16 block mode, the height and width are 1 / 4, 1 / 8 and 1 / 16 of <code>pstsrc1</code> respectively.	Output
pstThr	Output SAD thresholding image pointer. According to <code>pstSadCtrl→enOutCtrl</code> , if output is required, it cannot be empty. According to <code>pstSadCtrl→enMode</code> , corresponding to 4x4, 8x8 and 16x16 block mode, the height and width are 1 / 4, 1 / 8 and 1 / 16 of <code>pstSrc1</code> respectively.	Output
pstCtrl	Control information pointer. Cannot be empty.	Input
bInstant	Return result flag in time.	Input

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	U8C1	1byte	64x64~1920x1024
pstSrc2	U8C1	1byte	Same as pstSrc1
pstSad	U8C1、U16C1	16byte	According to pstSad-Ctrl→enMode, corresponding to 4x4, 8x8 and 16x16 block mode, the height and width are 1 / 4, 1 / 8 and 1 / 16 of pstSrc1 respectively.
pstThr	U8C1	16 byte	According to pstSad-Ctrl→enMode, corresponding to 4x4, 8x8 and 16x16 block mode, the height and width are 1 / 4, 1 / 8 and 1 / 16 of pstSrc1 respectively.

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.33 BufFlush

**【Description】**

For an image created using CVI\_IVE\_CreateImage\_Cached, before accessing the image content on IVE hardware,

this function must be used to update the cache data into RAM.

**【Syntax】**

```
CVI_S32 CVI_IVE_BufFlush(IVE_HANDLE pIveHandle, IVE_IMAGES_S *pstImg);
```

**【Parameter】** c

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstImg	the image content of the operation	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.34 BufRequest

**【Description】**

For the image created using `CVI_IVE_CreateImage_Cached`, before accessing the content pointed to by `u64VirAddr` on RISC-V

this function must be used to update the RAM content to cache.

**【Syntax】**

```
CVI_S32 CVI_IVE_BufRequest(IVE_HANDLE pIveHandle, IVE_IMAGES_S *pstImg);
```

**【Parameter】**

Parameter	Description	Input/Output
<code>pIveHandle</code>	the handle of the task	Input
<code>pstImg</code>	the image content of the operation	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `cvi_comm_ive.h` `cvi_ive.h`

## 3.35 CreateMemInfo

**【Description】**

Create a block of memory for use by the `IVE_MEM_S` structure.

**【Syntax】**

```
CVI_S32 CVI_IVE_CreateMemInfo(IVE_HANDLE pIveHandle, IVE_MEM_INFO_S *pstMemInfo,
↪ CVI_U32 u32ByteSize);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstMemInfo	the created memory structure. Cannot be empty.	Input
u32ByteSize	the byte volume of the created memory structure	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.36 CreatDataInfo

**【Description】**

Create a block of memory for use by the IVE\_DATA\_S structure.

**【Syntax】**

```
CVI_S32 CVI_IVE_CreateDataInfo(IVE_HANDLE pIveHandle, IVE_DATA_S *pstDataInfo, ↪
↪ CVI_U16 u16Width, CVI_U16 u16Height);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstDataInfo	the structure created of IVE_DATA_S. Cannot be empty	Input
u16Width	the width of Data	Input
u16Height	the height of Data	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.37 CreateImage

**【Description】**

Create an image memory for use.

The image created with this function will automatically map the contents of `u64PhyAddr` and `u64VirAddr`.

There is no need to Flush or Invalidate the cache.

**【Syntax】**

```
CVI_S32 CVI_IVE_CreateImage(IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstImg, IVE_
↪IMAGE_TYPE_E enType, CVI_U16 u16Width, CVI_U16 u16Height);
```

**【Parameter】**

Parameter	Description	Input/Output
<code>pIveHandle</code>	the handle of the task	Input
<code>pstImg</code>	Create a memory structure for the image.	Output
<code>enType</code>	the created image memory structure format	Input
<code>u16Width</code>	the width of image	Input
<code>u16Height</code>	the height of image	Input

**【Return Value】** c

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `cvi_comm_ive.h` `cvi_ive.h`

## 3.38 CreateImage with Cache

**【Description】**

Create an image memory for use.

Images created using this function need to update the contents of `u64PhyAddr` and `u64VirAddr` using `CVI_IVE_BufFlush` and `CVI_IVE_BufRequest`.

**【Syntax】**

```
CVI_S32 CVI_IVE_CreateImage_Cached(IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstImg,   
↪IVE_IMAGE_TYPE_E enType, CVI_U32 u32Width, CVI_U32 u32Height);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstImg	Create a memory structure for the image.	Output
enType	Created image memory structure format	Input
u32Width	the width of image	Input
u32Height	the height of image	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header `file:cvl_comm_ive.h` `cvl_ive.h`

## 3.39 ResetImage

**【Description】**

Fill the content of the Image with a specific value.

**【Syntax】**

```
CVI_S32 CVI_IVE_ResetImage(IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstImg, CVI_U8   
↪val);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstImg	the created memory structure for image	Output
val	the pre-filled value for the image	Input

**【Return Value】**



Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.40 ReadImageArray

**【Description】**

Read the image from the buffer.

**【Syntax】**

```
CVI_S32 CVI_IVE_ReadImageArray (IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstImage,
↪char *pBuffer, IVE_IMAGE_TYPE_E enType,
CVI_U16 u16Width, CVI_U16 u16Height);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstImage	the created memory structure for image	Output
pBuffer	Buffer	Input
enType	the created image memory structure	Input
u16Width	the width of image	Input
u16Height	the height of image	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.41 ReadMem

### 【Description】

Read into IVE\_DATA\_S structure from file.

### 【Syntax】

```
CVI_S32 CVI_IVE_ReadMem(IVE_HANDLE pIveHandle, IVE_MEM_INFO_S *pstMem, const ↵  
↵char *filename, CVI_U32 uSize);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstMem	the structure of the IVE_MEM_INFO_S	Output
filename	the path of the file	Input
u32Size	the size of the Mem	Input

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.42 ReadMemArray

### 【Description】

Read data from buffer into IVE\_MEM\_INFO\_S structure.

### 【Syntax】

```
CVI_S32 CVI_IVE_ReadMemArray (IVE_HANDLE pIveHandle, IVE_MEM_INFO_S *pstMem, ↵  
↵char *pBuffer, CVI_U32 uSize);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstMem	the structure of the IVE_MEM_INFO_S	Output
pBuffer	Buffer	Input
u32Size	the size of the Buffer	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files:cvi\_comm\_ive.h cvi\_ive.h

## 3.43 ReadData

**【Description】**

Read data from the file into IVE\_DATA\_S structure.

**【Syntax】**

```
CVI_S32 CVI_IVE_ReadData(IVE_HANDLE pIveHandle, IVE_DATA_S *pstData, const char_↵  
↵*filename, CVI_U16 u16Width, CVI_U16 u16Height);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstData	the structure of the IVE_DATA_S	Output
filename	the path of the file	Input
u16Width	the width of the Data	Input
u16Height	the height of the Data	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files:cvi\_comm\_ive.h cvi\_ive.h

## 3.44 ReadDataArray

### 【Description】

Read data from the buffer into IVE\_DATA\_S structure.

### 【Syntax】

```
CVI_S32 CVI_IVE_ReadDataArray (IVE_HANDLE pIveHandle, IVE_DATA_S *pstData, char *pBuffer, CVI_U16 u16Width, CVI_U16 u16Height);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstData	IVE_DATA_S 结构。	Output
pBuffer	Buffer	Input
u16Width	the width of the image	Input
u16Height	the height of the image	Input

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvl\_comm\_ive.h cvl\_ive.h

## 3.45 ReadImage

### 【Description】

Read an image from the file location.

### 【Syntax】

```
IVE_IMAGE_S CVI_IVE_ReadImage(IVE_HANDLE pIveHandle, const char *filename, IVE_IMAGE_TYPE_E enType);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the name of the Image file	Input
enType	the image format you want to get	

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.46 ReadRawImage

**【Description】**

Read an image from the file location.

**【Syntax】**

```
IVE_IMAGE_S CVI_IVE_ReadRawImage(IVE_HANDLE pIveHandle, const char *filename, IVE_IMAGE_TYPE_E enType, CVI_U16 u16Width, CVI_U16 u16Height);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the name of the Image file	Input
enType	The image format you want to get	Input
u16Width	the width of the image	Input
u16Height	the height of the image	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.47 WriteData

### 【Description】

Write the contents of IVE\_DATA\_S to the file location.

### 【Syntax】

```
CVI_S32 CVI_IVE_WriteData(IVE_HANDLE pIveHandle, const char *filename, IVE_DATA_
↪S *pstData);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the location and filename of the stored file	Input
pstData	the content to be stored	Output

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.48 WriteMem

### 【Description】

Write the contents of IVE\_MEM\_INFO\_S to the file location.

### 【Syntax】

```
CVI_S32 CVI_IVE_WriteData(IVE_HANDLE pIveHandle, const char *filename, IVE_MEM_
↪INFO_S *pstMem);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the location and filename of the stored file	Input
pstMem	the content to be stored	Output

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.49 WriteImage

**【Description】**

Write an PNG image into the file location.

**【Syntax】**

```
CVI_S32 CVI_IVE_WriteImage(IVE_HANDLE pIveHandle, const char *filename, IVE_
↪ IMAGE_S *pstImg);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the location and filename of the stored file	Input
pstImg	the image content to be stored	Output

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.50 WriteRawImage

**【Description】**

Write an image into the file location.

**【Syntax】**

```
CVI_S32 CVI_IVE_WriteImg(IVE_HANDLE pIveHandle, const char *filename, IVE_IMAGE_
↪ S *pstImg);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
filename	the location and filename of the stored file	Input
pstImg	the image content to be stored	Output

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.51 Reset Register

**【Description】**

Reset the cache of IVE to its default value.

**【Syntax】**

```
CVI_S32 CVI_IVE_RESET(IVE_HANDLE pIveHandle, int select);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
select	the IVE Module to be reset	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`



## 3.52 Dump Register

### 【Description】

Output the cache values of IVE to the log.

### 【Syntax】

```
CVI_S32 CVI_IVE_DUMP(IVE_HANDLE pIveHandle);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.53 Split DiffFg of BgModel

### 【Description】

Extract DiffFg from the result of BgModel and store it as a YUV image.

### 【Syntax】

```
CVI_S32 CVI_IVE_DiffFg_Split(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstDiffFg, IVE_DST_IMAGE_S *pstBGDiffFg, IVE_DST_IMAGE_S *pstFrmDiffFg);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstDiffFg		Input
pstBGDiffFg		Output
pstFrmDiffFg		Output

### 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.54 Split ChgSta of BgModel

**【Description】**

Extract ChgSta from the result of BgModel and store it as a YUV image.

**【Syntax】**

```
CVI_S32 CVI_IVE_DiffFg_Split(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstChgSta, IVE_DST_IMAGE_S *pstChgStaImg, IVE_DST_IMAGE_S *pstChgStaFg, IVE_DST_IMAGE_S *pstChStaLift);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstChgSta		Input
pstChgStaImg		Output
pstChgStaFg		Output
pstChStaLift		Output

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.55 Query Tasks

**【Description】**

Query the current status of the Task.

**【Syntax】**

```
CVI_S32 CVI_IVE_QUERY(IVE_HANDLE pIveHandle, CVI_BOOL *pbFinish, CVI_BOOL *pbBlock);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pbFinish	Return whether the task has ended.	Output
bBlock	True indicates blocked task	Output

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.56 Image2VideoFrameInfo

**【Description】**

Convert IVE image format to Video Frame Info format.

**【Syntax】**

```
CVI_S32 CVI_IVE_Image2VideoFrameInfo(IVE_IMAGE_S *pstIISrc,  
VIDEO_FRAME_INFO_S *pstVFIDst);
```

**【Parameter】**

Parameter	Description	Input/Output
pstIISrc	the Inputted image content	Input
pstVFIDst	the outputted image content	Output

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.57 VideoFrameInfo2Image

### 【Description】

Convert Video Frame Info image format to IVE format.

### 【Syntax】

```
CVI_S32 CVI_IVE_VideoFrameInfo2Image(VIDEO_FRAME_INFO_S *pstVFISrc, IVE_IMAGE_S *pstIIDst);
```

### 【Parameter】

Parameter	Description	Input/Output
pstIISrc	the Inputted image content	Input
pstVFIDst	the outputted image content	Output

### 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.58 FreeM

### 【Description】

Release an IVE\_MEM\_INFO\_S structure.

### 【Syntax】

```
CVI_S32 CVI_SYS_FreeM(IVE_HANDLE pIveHandle, IVE_MEM_INFO_S *pstMem);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstMem	the memory structure to be released	Input

### 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `cvl_comm_ive.h` `cvl_ive.h`

## 3.59 FreeI

**【Description】**

Release an IVE\_IMAGE\_S structure.

**【Syntax】**

```
CVI_S32 CVI_SYS_FreeI(IVE_HANDLE pIveHandle, IVE_IMAGE_S *pstImg);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstImg	the inputted image	Input

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header file: `cvl_comm_ive.h` `cvl_ive.h`

## 3.60 FreeD

**【Description】**

Release an IVE\_DATA\_S structure.

**【Syntax】**

```
CVI_S32 CVI_SYS_FreeD(IVE_HANDLE pIveHandle, IVE_DATA_S *pstData);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstData	the inputted Data	Input

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.61 Thresh\_S16

**【Description】**

Create thresholding task from S16 data to 8-bit data.

**【Syntax】**

```
CVI_S32 CVI_IVE_Thresh_S16(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_THRESH_S16_CTRL_S *pstThrS16Ctrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc	the inputted image's pointer. Cannot be empty.	Input
pstDst	the outputted image's pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Threshold parameter structure pointer, cannot be empty.	Input
bInstant	the reference value	Output

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.62 Thresh\_U16

### 【Description】

Create thresholding task from U16 data to 8-bit data.

### 【Syntax】

```
CVI_S32 CVI_IVE_Thresh_U16(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_
↪ DST_IMAGE_S *pstDst, IVE_THRESH_U16_CTRL_S *pstCtrl, CVI_BOOL bInstant);
```

### 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc	the inputted image' s pointer. Cannot be empty.	Input
pstDst	the outputted image' s pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	Threshold parameter structure pointer, cannot be empty.	Input
bInstant	the reference value	Output

### 【Return Value】

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

### 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.63 Resize

### 【Description】

Create an image Resize task that supports both Bilinear Interpolation and Area Interpolation methods.

### 【Syntax】

```
CVI_S32 CVI_IVE_Resize(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S
astSrc[], IVE_DST_IMAGE_S astDst[], IVE_RESIZE_CTRL_S *pstCtrl, CVI_BOOL
↪ bInstant);
```

## 【Parameter】

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
astSrc	The inputted image array. Cannot be empty.	Input
astDst	The outputted image array. Cannot be empty. The type of the image must be the same as astSrc.	Output
pstCtrl	Threshold parameter structure pointer, cannot be empty.	Input
bInstant	the reference value	Output

Parameter	Support Image Type	Address Alignment	Resolution
astSrc	U8C1 or U8C3_PLANAR	1byte	
astDst	U8C1 or U8C3_PLANAR	1byte	

## 【Return Value】

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

## 【Requirement】

- Header files: cvi\_comm\_ive.h cvi\_ive.h

## 3.64 16BitTo8Bit

## 【Description】

Create a linearization task for converting 16-bit image data to 8-bit image data.

## 【Syntax】

```
CVI_S32 CVI_IVE_16BitTo8Bit (IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc, IVE_DST_IMAGE_S *pstDst, IVE_16BIT_TO_8BIT_CTRL_S *pstCtrl, bool bInstant);
```

## 【Parameter】



Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc	The inputted image pointer. Cannot be empty.	Input
pstDst	The outputted image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	The threshold value parameter structure pointer. Cannot be empty.	Input
bInstant	the reference value	Output

**【Return Value】**

Return Value	Description
0	Success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.65 RGB YUV Erode to Dilate

**【Description】****【Syntax】**

```
CVI_S32 CVI_IVE_rgbPYuvToErodeToDilate(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S  
↪ *pstSrc, IVE_DST_IMAGE_S *pstDst1, IVE_DST_IMAGE_S *pstDst2, IVE_FILTER_CTRL_S  
↪ *pstCtrl, CVI_BOOL  
bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc	The inputted image pointer. Cannot be empty.	Input
pstDst1	The outputted image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstDst2	The outputted image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	The threshold value parameter structure pointer. Cannot be empty.	Input
bInstant	the reference value	Output

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.66 STCandiCorner

**【Description】**

Compute candidate corner points.

**【Syntax】**

```
CVI_S32 CVI_IVE_STCandiCorner(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc,
↪ IVE_DST_IMAGE_S *pstDst, IVE_ST_CANDI_CORNER_CTRL_S *pstCtrl, CVI_BOOL
↪ bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc	The inputted image pointer. Cannot be empty.	Input
pstDst	The outputted image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	The threshold value parameter structure pointer. Cannot be empty.	Input
bInstant	the reference value	Output

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

## 3.67 Background Subtraction

**【Description】**

Create a background subtraction task.

**【Syntax】**

```
CVI_S32 CVI_IVE_FrameDiffMotion(IVE_HANDLE pIveHandle, IVE_SRC_IMAGE_S *pstSrc1,  
↪ IVE_SRC_IMAGE_S *pstSrc2, IVE_DST_IMAGE_S *pstDst, IVE_FRAME_DIFF_MOTION_CTRL_  
↪ S *pstCtrl, CVI_BOOL bInstant);
```

**【Parameter】**

Parameter	Description	Input/Output
pIveHandle	the handle of the task	Input
pstSrc1	The inputted image pointer. Cannot be empty.	Input
pstSrc2	The inputted image pointer. Cannot be empty.	Input
pstDst	The outputted image pointer. Cannot be empty. The height and width are the same as pstSrc.	Output
pstCtrl	The threshold value parameter structure pointer. Cannot be empty.	Input
bInstant	the reference value	Output

Parameter	Support Image Type	Address Alignment	Resolution
pstSrc1	U8C1	16 byte	64x64~1920x1024
pstSrc2	U8C1	16 byte	Same as pstSrc1
pstDst	U8C1	16 byte	Same as pstSrc1

**【Return Value】**

Return Value	Description
0	success
Non 0	Failure, please refer to the error code.

**【Requirement】**

- Header files: `cvi_comm_ive.h` `cvi_ive.h`

# 4 Data Type and Data Structure

---

The definitions of IVE related data types and data structures are as follows:

- `IVE_IMAGE_TYPE_E` Define the image types supported by 2D generalized image.
- `IVE_IMAGE_S` :Define the information of two dimensional generalized image.
- `IVE_SRC_IMAGE_S` :Define the source image.
- `IVE_DST_IMAGE_S` :Define the output image.
- `IVE_DATA_S` :Define two-dimensional image information in bytes.
- `IVE_SRC_DATA_S` :Define two-dimensional source data information in bytes.
- `IVE_DST_DATA_S` :Define the two-dimensional output data information in bytes.
- `IVE_MEM_INFO_S` :Define one-dimensional data memory information.
- `IVE_SRC_MEM_INFO_S` : Define one-dimensional source data.
- `IVE_DST_MEM_INFO_S` :Define one-dimensional output data.
- `IVE_8BIT_U` :Define an 8-bit data union.
- `IVE_DMA_MODE_E` :Define DMA operation mode.
- `IVE_DMA_CTRL_S` :Define DMA control information.
- `IVE_FILTER_CTRL_S` :Define the template filter control information.
- `IVE_CSC_MODE_E` :Define the color space conversion mode.
- `IVE_CSC_CTRL_S` :Define color space conversion control information.
- `IVE_SOBEL_OUT_CTRL_E` :Define Sobel output control information.
- `IVE_SOBEL_CTRL_S` :Define Sobel edge extraction control information.
- `IVE_MAG_AND_ANG_OUT_CTRL_E` :Define the output format of amplitude and angle calculation.
- `IVE_MAG_AND_ANG_CTRL_S` :Define the control information of amplitude and phase calculation.
- `IVE_MAG_DIST_E` :Define the method for calculating gradient magnitude distance.
- `IVE_DILATE_CTRL_S` :Define dilation control information.
- `IVE_ERODE_CTRL_S` :Define erosion control information.
- `IVE_BLOCK_CTRL_S` :Define IVE\_Block control information.

- *IVE\_SUB\_MODE\_E* :Define the output format for subtracting two images.
- *IVE\_SUB\_CTRL\_S* :Define two image subtraction control Parameter.
- *IVE\_INTEG\_OUT\_CTRL\_E* :Define the output control Parameter of integral image.
- *IVE\_INTEG\_CTRL\_S* :Define the control parameter of calculating integral image.
- *IVE\_THRESH\_MODE\_E* :Define image binary output format.
- *IVE\_THRESH\_CTRL\_S* :Define image binary control information.
- *IVE\_THRESH\_S16\_MODE\_E* :Define the thresholding mode of 16 bit signed image.
- *IVE\_THRESH\_S16\_CTRL\_S* :Define the thresholding control parameter of 16 bit signed image.
- *IVE\_THRESH\_U16\_MODE\_E* :Define the thresholding mode of 16 bit unsigned image.
- *IVE\_THRESH\_U16\_CTRL\_S* :Define the thresholding control parameter of 16 bit unsigned image.
- *IVE\_16BIT\_TO\_8BIT\_MODE\_E* :Define the conversion mode from 16 bit image to 8 bit image.
- *IVE\_16BIT\_TO\_8BIT\_CTRL\_S* :Define the conversion control parameter from 16 bit image to 8 bit image.
- *IVE\_ORD\_STAT\_FILTER\_MODE\_E* :Define the order statistics filtering mode.
- *IVE\_ORD\_STAT\_FILTER\_CTRL\_S* :Define the order statistics filter control parameter.
- *IVE\_EQUALIZE\_HIST\_CTRL\_S* :Define the histogram equalization control parameter.
- *IVE\_ADD\_CTRL\_S* :Define the weighted addition control parameters for two images.
- *IVE\_NCC\_DST\_MEM\_S* :Define the output memory information of NCC.
- *IVE\_LBP\_CMP\_MODE\_E* :Define LBP texture calculation control Parameter.
- *IVE\_LBP\_CTRL\_S* :Define LBP texture calculation control Parameter.
- *IVE\_NORM\_GRAD\_OUT\_CTRL\_E* :Define the task of calculating normalized gradient information and output the control enumeration type.
- *IVE\_NORM\_GRAD\_CTRL\_S* :Define the control parameter of calculating normalized gradient information.
- *IVE\_SAD\_MODE\_E* :Define SAD calculation mode.
- *IVE\_SAD\_OUT\_CTRL\_E* :Define SAD output control mode.
- *IVE\_SAD\_CTRL\_S* :Define SAD control parameter.
- *IVE\_RESIZE\_MODE\_E* :Define the mode of Resize.
- *IVE\_RESIZE\_CTRL\_S* :Define the resize control parameter.
- *IVE\_HOG\_CTRL\_S* :Define and calculate HOG (histogram of oriented gradient) feature control Parameter.
- *IVE\_GRAD\_FG\_CTRL\_S* : Define the Gradfg control parameter.
- *IVE\_GRAD\_FG\_MODE\_E* : Define the mode of Gradfg.

## 4.1 Define Data Types

### 【Description】

Define fixed-point data types.

### 【Syntax】

Shared with middleware, see `cvi_type.h` for details.

## 4.2 Define Structure Type

### 4.2.1 IVE\_IMAGE\_TYPE\_E\_NUM

### 【Description】

Define the image types supported by 2D generalized image.

### 【Syntax】

```
typedef enum IVE_IMAGE_TYPE {  
  
    IVE_IMAGE_TYPE_U8C1 = 0x0,  
  
    IVE_IMAGE_TYPE_S8C1 = 0x1,  
  
    IVE_IMAGE_TYPE_YUV420SP = 0x2,  
  
    IVE_IMAGE_TYPE_YUV422SP = 0x3,  
  
    IVE_IMAGE_TYPE_YUV420P = 0x4,  
  
    IVE_IMAGE_TYPE_YUV422P = 0x5,  
  
    IVE_IMAGE_TYPE_S8C2_PACKAGE = 0x6,  
  
    IVE_IMAGE_TYPE_S8C2_PLANAR = 0x7,  
  
    IVE_IMAGE_TYPE_S16C1 = 0x8,  
  
    IVE_IMAGE_TYPE_U16C1 = 0x9,  
  
    IVE_IMAGE_TYPE_U8C3_PACKAGE = 0xa,  
  
    IVE_IMAGE_TYPE_U8C3_PLANAR = 0xb,  
  
    IVE_IMAGE_TYPE_S32C1 = 0xc,  
  
    IVE_IMAGE_TYPE_U32C1 = 0xd,  
  
}
```

(continues on next page)

(continued from previous page)

```
IVE_IMAGE_TYPE_S64C1 = 0xe,  
IVE_IMAGE_TYPE_U64C1 = 0xf,  
IVE_IMAGE_TYPE_BF16C1 = 0x10,  
IVE_IMAGE_TYPE_FP32C1 = 0x11,  
IVE_IMAGE_TYPE_BUTT  
} IVE_IMAGE_TYPE_E;
```

**【Member】**



Member	Description
IVE_IMAGE_TYPE_U8C1	Each pixel is represented by a single-channel image with 8-bit unsigned data. See Figure 1-2.
IVE_IMAGE_TYPE_S8C1	Each pixel is represented by a single-channel image with 8 bit signed data. See Figure 1-2.
IVE_IMAGE_TYPE_YUV420SP	YUV420 semiplanar format image. See Figure 1-3.
IVE_IMAGE_TYPE_YUV422SP	YUV422 semiplanar format image. See Figure 1-4.
IVE_IMAGE_TYPE_YUV420P	YUV420 planar format image. See Figure 1-5.
IVE_IMAGE_TYPE_YUV422P	YUV422 planar format image. See Figure 1-6.
IVE_IMAGE_TYPE_S8C2_PACKAGE	Each pixel is represented by two 8bit signed data, and two-channel images are stored in package format. See Figure 1-7.
IVE_IMAGE_TYPE_S8C2_PLANAR	Each pixel is represented by two 8bit signed data, and two-channel images are stored in planar format. See Figure 1-8.
IVE_IMAGE_TYPE_S16C1	Each pixel uses a 16 bit signed data to represent a single-channel image. See Figure 1-2.
IVE_IMAGE_TYPE_U16C1	Each pixel uses a 16 bit unsigned data to represent a single-channel image. See Figure 1-2.
IVE_IMAGE_TYPE_U8C3_PACKAGE	Each pixel is represented by three 8-bit unsigned data, and three-channel images are stored in package format. See Figure 1-9.
IVE_IMAGE_TYPE_U8C3_PLANAR	Each pixel uses three 8-bit unsigned data to represent a three-channel image of one pixel, which is stored in planar format. See Figure 1-10.
IVE_IMAGE_TYPE_S32C1	Each pixel uses a 32 bit signed data to represent a single-channel image. See Figure 1-2.
IVE_IMAGE_TYPE_U32C1	Each pixel represents a single channel image with 32-bit unsigned data. See Figure 1-2.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_IMAGE\_S
- IVE\_SRC\_IMAGE\_S
- IVE\_DST\_IMAGE\_S

## 4.2.2 IVE\_IMAGE\_S

**【Description】**

Define the information of two-dimensional generalized image.

**【Syntax】**

```
typedef struct IVE_IMAGE
{
    IVE_IMAGE_TYPE_E enType;

    CVI_U64 u64phyAddr[3];

    CVI_U64 u64VirAddr[3];

    CVI_U32 u32Stride[3];

    CVI_U32 u32Width;

    CVI_U32 u32Height;

    CVI_U32 u32Reserved;
} IVE_IMAGE_S;
```

**【Member】**

Member	Description
enType	The image type of generalized image.
U64phyAddr	Physical address array of generalized image.
u64VirAddr	Virtual address array of generalized image.
u32Stride	The stride of generalized image.
u32Width	The width of the generalized image.
u32Height	The height of the generalized image.
u32Reserved	Reserved bit.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_IMAGE\_TYPE\_E
- IVE\_SRC\_IMAGE\_S
- IVE\_DST\_IMAGE\_S

### 4.2.3 IVE\_SRC\_IMAGE\_S

**【Description】**

Define the source image.

**【Syntax】**

```
typedef IVE_IMAGE_S IVE_SRC_IMAGE_S;
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_IMAGE\_S
- IVE\_DST\_IMAGE\_S

### 4.2.4 IVE\_DST\_IMAGE\_S

**【Description】**

Define the output image.

**【Syntax】**

```
typedef IVE_IMAGE_S IVE_DST_IMAGE_S;
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_IMAGE\_S
- IVE\_SRC\_IMAGE\_S

## 4.2.5 IVE\_DATA\_S

### 【Description】

Defines two-dimensional data information in bytes.

### 【Syntax】

```
typedef struct _IVE_DATA_S
{
    IVE_IMAGE_TYPE_E enType;

    CVI_U64 u64PhyAddr;

    CVI_U64 u64VirAddr;

    CVI_U32 u32Stride;

    CVI_U32 u32Width;

    CVI_U32 u32Height;

    CVI_U32 u32Reserved;
} IVE_DATA_S;
```

### 【Member】

Member	Description
u64PhyAddr	Physical address array of generalized image.
u64VirAddr	Virtual address array of generalized image.
u32Stride	The stride of generalized image.
u32Width	The width of the generalized image.
u32Height	The height of the generalized image.
u32Reserved	Reserved bit.

### 【Note】

None.

### 【Related Data Type and Interface】

None.

## 4.2.6 IVE\_SRC\_DATA\_S

**【Description】**

Define two-dimensional source data information in bytes.

**【Syntax】**

```
typedef IVE_DATA_S IVE_SRC_DATA_S
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_IMAGE\_S

IVE\_DST\_DATA\_S

## 4.2.7 IVE\_DST\_DATA\_S

**【Description】**

Define the two-dimensional output data information in bytes.

**【Syntax】**

```
typedef IVE_DATA_S IVE_DST_DATA_S
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_IMAGE\_S

IVE\_SRC\_IMAGE\_S

## 4.2.8 IVE\_MEM\_INFO\_S

**【Description】**

Define one-dimensional data memory information.

**【Syntax】**

```
typedef struct _IVE_MEM_INFO_S
{
    CVI_U64 u64PhyAddr;

    CVI_U64 u64VirAddr;

    CVI_U32 u32Size;
} IVE_MEM_INFO_S;
```

**【Member】**

Member	Description
u64PhyAddr	The physical address of one-dimensional data.
u64VirAddr	The virtual address of one-dimensional data.
u32Size	The byte number of one-dimensional data.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SRC\_MEM\_INFO\_S

IVE\_DST\_MEM\_INFO\_S

## 4.2.9 IVE\_SRC\_MEM\_INFO\_S

**【Description】**

Define one-dimensional source data.

**【Syntax】**

```
typedef IVE_MEM_INFO_S IVE_SRC_MEM_INFO_S;
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_MEM\_INFO\_S

IVE\_DST\_MEM\_INFO\_S

## 4.2.10 IVE\_DST\_MEM\_INFO\_S

**【Description】**

Define one-dimensional source data.

**【Syntax】**

```
typedef IVE_MEM_INFO_S IVE_DST_MEM_INFO_S;
```

**【Member】**

None.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_MEM\_INFO\_S

IVE\_SRC\_MEM\_INFO\_S

## 4.2.11 IVE\_8BIT\_U

**【Description】**

Define an 8-bit data union.

**【Syntax】**

```
typedef union _IVE_8BIT
{
    CVI_S8 s8Val;

    CVI_U8 u8Val;
} IVE_8BIT_U;
```

**【Member】**

Member	Description
s8Val	Signed 8bit value.
u8Val	Unsigned 8 bit value.

**【Note】**

None.

**【Related Data Type and Interface】**

None.

## 4.2.12 IVE\_POINT\_U16\_S

**【Description】**

Define a data structure for unsigned 16-bit coordinate.

**【Syntax】**

```
typedef struct _IVE_POINT_U16_S
{
    CVI_U16 u16X;

    CVI_U16 u16Y;
} IVE_POINT_U16_S;
```

**【Member】**

Member	Description
u16X	Unsigned 16bit X coordinate.
u16Y	Unsigned 16bit Y coordinate.

**【Note】**

None.

**【Related Data Type and Interface】**

None.

## 4.2.13 IVE\_POINT\_S16\_S

**【Description】**

Define a data structure for signed 16-bit coordinate.

**【Syntax】**

```
typedef struct _IVE_POINT_S16_S
{
```

(continues on next page)



(continued from previous page)

```
CVI_S16 s16X;

CVI_S16 s16Y;

} IVE_POINT_S16_S;
```

【Member】

Member	Description
s16X	The signed 16bit X coordinate.
s16Y	The signed 16bit Y coordinate.

【Note】

None.

【Related Data Type and Interface】

None.

4.2.14 IVE\_DMA\_MODE\_E

【Description】

Define DMA operation mode.

【Syntax】

```
typedef struct IVE_DMA_MODE
{
    IVE_DMA_MODE_DIRECT_COPY = 0x0,
    IVE_DMA_MODE_INTERVAL_COPY = 0x1,
    IVE_DMA_MODE_SET_3BYTE = 0x2,
    IVE_DMA_MODE_SET_8BYTE = 0x3,
    IVE_DMA_MODE_BUTT
} IVE_DMA_MODE_E;
```

【Member】

Member	Description
IVE_DMA_MODE_DIRECT_COPY	The mode of direct quick copy.
IVE_DMA_MODE_INTERVAL_COPY	The mode of interval copy mode, more details to see CVI_IVE_DMA <b>【Note】</b> Description
IVE_DMA_MODE_SET_3BYTE	The mode of 3 byte assignment, see CVI_IVE_DMA <b>【Note】</b> Description
IVE_DMA_MODE_SET_8BYTE	The mode of 8 byte assignment, more details to see CVI_IVE_DMA <b>【Note】</b> Description

**【Note】**

None.

**【Related Data Type and Interface】**

None.

## 4.2.15 IVE\_DMA\_CTRL\_S

**【Description】**

Define DMA control information.

**【Syntax】**

```
typedef struct IVE_DMA_CTRL
{
    IVE_DMA_MODE_E enMode;

    CVI_U64 u64Val;

    CVI_U8 u8HorSegSize;

    CVI_U8 u8ElemSize;

    CVI_U8 u8VerSegRows;
}IVE_DMA_CTRL_S;
```

**【Member】**

Member	Description
enMode	DMA operation mode.
u64Val	Only for the assignment mode which is used for memory assignment, and the 3byte assignment mode is saved by lower 3byte.
u8HorSegSize	Only for the interval copy mode. Segment size of horizontally dividing the source image into rows. Value range: {2, 3, 4, 8, 16}.
u8ElemSize	Only for the interval copy mode, and the first u8ElemSizebyte in each segment is the valid copy field. Value range: [1, u8HorSegSize-1].
u8VerSegRows	Only for the interval copy mode, the first row of data in each u8VerSegRows row is divided into u8HorSegSize segments, and the first u8ElemSize bytes in each segment are copied

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_DMA\_MODE\_E

## 4.2.16 IVE\_FILTER\_CTRL\_S

**【Description】**

Define the template filter control information.

**【Syntax】**

```
typedef struct IVE_FILTER_CTRL
{
    CVI_S8 as8Mask[25];

    CVI_U8 u8Norm;
} IVE_FILTER_CTRL_S;
```

**【Member】**

Member	Description
enMode	5x5 template coefficient, peripheral coefficient is set to 0 to realize 3x3 template filtering.
u8Norm	Normalization Parameter. Value range: [0, 13].

**【Note】**

Different filtering effects can be achieved by configuring different template coefficients.

**【Related Data Type and Interface】**

None.

## 4.2.17 IVE\_CSC\_MODE\_E

**【Description】**

Define the color space conversion mode.

**【Syntax】**

```
typedef enum IVE_CSC_MODE_E
{
    IVE_CSC_MODE_VIDEO_BT601_YUV2RGB = 0x0,
    IVE_CSC_MODE_VIDEO_BT709_YUV2RGB = 0x1,
    IVE_CSC_MODE_PIC_BT601_YUV2RGB = 0x2,
    IVE_CSC_MODE_PIC_BT709_YUV2RGB = 0x3,
    IVE_CSC_MODE_PIC_BT601_YUV2HSV = 0x4,
    IVE_CSC_MODE_PIC_BT709_YUV2HSV = 0x5,
    IVE_CSC_MODE_PIC_BT601_YUV2LAB = 0x6,
    IVE_CSC_MODE_PIC_BT709_YUV2LAB = 0x7,
    IVE_CSC_MODE_VIDEO_BT601_RGB2YUV = 0x8,
    IVE_CSC_MODE_VIDEO_BT709_RGB2YUV = 0x9,
```

(continues on next page)

(continued from previous page)

```

IVE_CSC_MODE_PIC_BT601_RGB2YUV = 0xa,

IVE_CSC_MODE_PIC_BT709_RGB2YUV = 0xb,

IVE_CSC_MODE_BUTT

} IVE_CSC_MODE_E;

```

**【Member】**

Member	Description
IVE_CSC_MODE_VIDEO_BT601_YUV2RGB	BT601 YUV2RGB Video format conversion
IVE_CSC_MODE_VIDEO_BT709_YUV2RGB	BT709 YUV2RGB Video format conversion
IVE_CSC_MODE_PIC_BT601_YUV2RGB	BT601 YUV2RGB Video format conversion
IVE_CSC_MODE_PIC_BT709_YUV2RGB	BT709 YUV2RGB Video format conversion
IVE_CSC_MODE_PIC_BT601_YUV2HSV	BT601 YUV2HSV Video format conversion
IVE_CSC_MODE_PIC_BT709_YUV2HSV	BT709 YUV2HSV Video format conversion
IVE_CSC_MODE_PIC_BT601_YUV2LAB	BT601 YUV 2LAB Video format conversion
IVE_CSC_MODE_PIC_BT709_YUV2LAB	BT709 YUV 2LAB Video format conversion
IVE_CSC_MODE_VIDEO_BT601_RGB2YUV	BT601 RGB2YUV Video 格式转换
IVE_CSC_MODE_VIDEO_BT709_RGB2YUV	BT709 RGB2YUV Video format conversion
IVE_CSC_MODE_PIC_BT601_RGB2YUV	BT601 RGB2YUV Video format conversion
IVE_CSC_MODE_PIC_BT709_RGB2YUV	BT709 RGB2YUV Video format conversion

**【Note】****【Related Data Type and Interface】**

IVE\_CSC\_CTRL\_S

## 4.2.18 IVE\_CSC\_CTRL\_S

**【Description】**

Define color space conversion control information.

**【Syntax】**

```

typedef struct cviIVE_CSC_CTRL_S
{

    IVE_CSC_MODE_E.
    enMode;

}IVE_CSC_CTRL_S;

```

**【Member】**

Member	Description
enMode	Working mode

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_CSC\_MODE\_E

## 4.2.19 IVE\_SOBEL\_OUT\_CTRL\_E

**【Description】**

Define Sobel output control information.

**【Syntax】**

```
typedef enum IVE_SOBEL_OUT_CTRL
{
    IVE_SOBEL_OUT_CTRL_BOTH = 0x0,
    IVE_SOBEL_OUT_CTRL_HOR = 0x1,
    IVE_SOBEL_OUT_CTRL_VER = 0x2,
    IVE_SOBEL_OUT_CTRL_BUTT
} IVE_SOBEL_OUT_CTRL_E;
```

**【Member】**

Member	Description
IVE_SOBEL_OUT_CTRL_BOTH	Output the results of filtering with both the template and the transpose template simultaneously.
IVE_SOBEL_OUT_CTRL_HOR	Only output the result of filtering with the template.
IVE_SOBEL_OUT_CTRL_VER	Only output the result of filtering with the transpose template.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SOBEL\_CTRL\_S

### 4.2.20 IVE\_SOBEL\_CTRL\_S

**【Description】**

The Sobel like gradient is defined to calculate the control information.

**【Syntax】**

```
typedef struct IVE_SOBEL_CTRL
{
    IVE_SOBEL_OUT_CTRL_E enOutCtrl;

    CVI_S8 as8Mask[25];
} IVE_SOBEL_CTRL_S;
```

**【Member】**

Member	Description
enOutCtrl	Output control enumeration parameter.
U8MaskSize	Mask Size
as8Mask[25]	Template coefficient.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SOBEL\_OUT\_CTRL\_E

### 4.2.21 IVE\_MAG\_AND\_ANG\_OUT\_CTRL\_E

**【Description】**

The output format of gradient amplitude and angle calculation is defined.

**【Syntax】**

```
typedef struct IVE_MAG_AND_ANG_OUT_CTRL
{
    IVE_MAG_AND_ANG_OUT_CTRL_MAG = 0x0,

    IVE_MAG_AND_ANG_OUT_CTRL_MAG_AND_ANG = 0x1,

    IVE_MAG_AND_ANG_OUT_CTRL_BUTT

} IVE_MAG_AND_ANG_OUT_CTRL_E;
```

**【Member】**

Member	Description
IVE_MAG_AND_ANG_OUT_CTRL_MAG	Only output amplitude.
IVE_MAG_AND_ANG_OUT_CTRL_MAG_AND_ANG	Output Amplitude and angle value simultaneously.

## 4.2.22 IVE\_MAG\_AND\_ANG\_CTRL\_S

**【Description】**

Define the control information of amplitude and phase calculation.

**【Syntax】**

```
typedef struct IVE_MAG_AND_ANG_CTRL
{
    IVE_MAG_AND_ANG_OUT_CTRL_E enOutCtrl;

    CVI_U16 u16Thr;

    CVI_S8 as8Mask[25];
} IVE_MAG_AND_ANG_OUT_CTRL_S;
```

**【Member】**

Member	Description
enOutCtrl	The output format
u16Thr	The threshold value
as8Mask	5x5 Filter

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_MAG\_AND\_ANG\_OUT\_CTRL\_E



### 4.2.23 IVE\_DILATE\_CTRL\_S

**【Description】**

Define dilation control information.

**【Syntax】**

```
typedef struct _IVE_DILATE_CTRL_S
{
    CVI_U8 au8Mask[25];
} IVE_DILATE_CTRL_S;
```

**【Member】**

Member	Description
au8Mask[25]	5x5 template coefficient. Value range: 0 or 255

**【Note】**

None.

**【Related Data Type and Interface】**

None.

### 4.2.24 IVE\_ERODE\_CTRL\_S

**【Description】**

Define erosion control information.

**【Syntax】**

```
typedef IVE_DILATE_CTRL_S IVE_ERODE_CTRL_S;
```

**【Member】**

Member	Description
au8Mask[25]	5x5 template coefficient. Value range: 0 or 255

**【Note】**

None.

**【Related Data Type and Interface】**

None.

### 4.2.25 IVE\_THRESH\_MODE\_E

**【Description】**

Define image binary output format.

**【Syntax】**

```
typedef enum IVE_THRESH_MODE
{
    IVE_THRESH_MODE_BINARY = 0x0,
    IVE_THRESH_MODE_TRUNC = 0x1,
    IVE_THRESH_MODE_TO_MINVAL = 0x2,
    IVE_THRESH_MODE_MIN_MID_MAX = 0x3,
    IVE_THRESH_MODE_ORI_MID_MAX = 0x4,
    IVE_THRESH_MODE_MIN_MID_ORI = 0x5,
    IVE_THRESH_MODE_MIN_ORI_MAX = 0x6,
    IVE_THRESH_MODE_ORI_MID_ORI = 0x7,
} IVE_THRESH_MODE_E;
```

**【Member】**

Member	Description
IVE_THRESH_MODE_BINARY	srcVal < lowThr, dstVal = minVal; srcVal > lowThr, dstVal = maxVal。
IVE_THRESH_MODE_TRUNC	srcVal < lowThr, dstVal = srcVal srcVal > lowThr, dstVal = maxVal
IVE_THRESH_MODE_TO_MINVAL	srcVal < lowThr, dstVal = minVal srcVal > lowThr, dstVal = srcVal
IVE_THRESH_MODE_MIN_MID_MAX	srcVal < lowThr, dstVal = minVal lowThr < srcVal < highThr, dstVal = midVal srcVal > highThr, dstVal = maxVal
IVE_THRESH_MODE_ORI_MID_MAX	srcVal < lowThr, dstVal = srcVal lowThr < srcVal < highThr dstVal = midVal srcVal > highThr, dstVal = maxVal
IVE_THRESH_MODE_MIN_MID_ORI	srcVal < lowThr, dstVal = minVal lowThr < srcVal < highThr dstVal = midVal srcVal > highThr, dstVal = srcVal
IVE_THRESH_MODE_MIN_ORI_MAX	srcVal < lowThr, dstVal = minVal lowThr < srcVal < highThr dstVal = srcVal srcVal > highThr, dstVal = maxVal
IVE_THRESH_MODE_ORI_MID_ORI	srcVal < lowThr, dstVal = srcVal lowThr < srcVal < highThr dstVal = midVal srcVal > highThr, dstVal = srcVal

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_CTRL\_S

## 4.2.26 IVE\_THRESH\_CTRL\_S

**【Description】**

Define image binary control information.

**【Syntax】**

```
typedef struct IVE_THRESH_CTRL
{
    CVI_U32 enMode;

    CVI_U8 u8LowThr;

    CVI_U8 u8HighThr;

    CVI_U8 u8MinVal;
```

(continues on next page)

(continued from previous page)

```

    CVI_U8 u8MidVal;

    CVI_U8 u8MaxVal;
}IVE_THRESH_CTRL_S;

```

**【Member】**

Member	Description
enMode	Thresholding operation mode.
u8LowThr	Low threshold. value range: [0,255]。
u8HighThr	High threshold. value range: [0,255]。
u8MinVal	Minimum value range: [0,255]。
u8MidVal	Median value range: [0,255]。
u8MaxVal	Maximum value range: [0,255]。

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_MODE\_E

### 4.2.27 IVE\_SUB\_MODE\_E

**【Description】**

Define the output format for subtracting two images.

**【Syntax】**

```
Typedef enum _IVE_SUB_MODE_E
{
    IVE_SUB_MODE_ABS = 0x0,
    IVE_SUB_MODE_SHIFT = 0x1,
    IVE_SUB_MODE_BUTT
} IVE_SUB_MODE_E;
```

**【Member】**

Member	Description
IVE_SUB_MODE_ABS	Subtract and take the absolute value.
IVE_SUB_MODE_SHIFT	Right shift the result by one bit and output, preserving the sign bit.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SUB\_CTRL\_S

### 4.2.28 IVE\_SUB\_CTRL\_S

**【Description】**

Define two image subtraction control Parameter.

**【Syntax】**

```
Typedef struct IVE_SUB_CTRL
{
    IVE_SUB_MODE_E enMode;
} IVE_SUB_CTRL_S;
```

**【Member】**

Member	Description
enMode	Two images subtraction mode

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SUB\_MODE\_E

## 4.2.29 IVE\_INTEG\_OUT\_CTRL\_E

**【Description】**

Define the output control Parameter of integral image.

**【Syntax】**

```
Typedef enum _IVE_INTEG_OUT_CTRL_E
{
    IVE_INTEG_OUT_CTRL_COMBINE = 0x0,
    IVE_INTEG_OUT_CTRL_SUM = 0x1,
    IVE_INTEG_OUT_CTRL_SQSUM = 0x2,
    IVE_INTEG_OUT_CTRL_BUTT
} IVE_INTEG_OUT_CTRL_E;
```

**【Member】**

Member	Description
IVE_INTEG_OUT_CTRL_COMBINE	Combined output of sum, square sum and integral image.
IVE_INTEG_OUT_CTRL_SUM	Only output sum integral image.
IVE_INTEG_OUT_CTRL_SQSUM	Only output the sum of squares and integral image.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_INTEG\_CTRL\_S

### 4.2.30 IVE\_INTEG\_CTRL\_S

**【Description】**

Define the control parameter of calculating integral image.

**【Syntax】**

```
Typedef struct _IVE_INTEG_CTRL_S
{
    IVE_INTEG_MODE_E enOutCtrl;
} IVE_INTEG_CTRL_S;
```

**【Member】**

Member	Description
enOutCtrl	The output control Parameter of the integral image.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_INTEG\_OUT\_CTRL\_E

### 4.2.31 IVE\_THRESH\_S16\_MODE\_E

**【Description】**

Define the thresholding mode of 16 bit signed image.

**【Syntax】**

```
typedef enum IVE_THRESH_S16_MODE_E
{
    IVE_THRESH_S16_MODE_S16_TO_S8_MIN_MID_MAX = 0x0,
    IVE_THRESH_S16_MODE_S16_TO_S8_MIN_ORI_MAX = 0x1,
    IVE_THRESH_S16_MODE_S16_TO_U8_MIN_MID_MAX = 0x2,
    IVE_THRESH_S16_MODE_S16_TO_U8_MIN_ORI_MAX = 0x3,
    IVE_INTEG_MODE_E enOutCtrl;
```

(continues on next page)

(continued from previous page)

```
} IVE_THRESH_S16_MODE_E;
```

**【Member】**

Member	Description
IVE_THRESH_S16_MODE_S16_TO_S8_MIN_VAL_LOW_THR	MIN_VAL_LOW_THR, dstVal = minVal; lowThr < srcVal highThr, dstVal = midVal; srcVal > highThr, dstVal = maxVal;
IVE_THRESH_S16_MODE_S16_TO_S8_MIN_VAL_MAX_THR	MIN_VAL_MAX_THR, dstVal = minVal; lowThr < srcVal highThr, dstVal = srcVal; srcVal > highThr, dstVal = maxVal;
IVE_THRESH_S16_MODE_S16_TO_U8_MIN_VAL_LOW_THR	MIN_VAL_LOW_THR, dstVal = minVal; lowThr < srcVal highThr, dstVal = midVal; srcVal > highThr, dstVal = maxVal;
IVE_THRESH_S16_MODE_S16_TO_U8_MIN_VAL_MAX_THR	MIN_VAL_MAX_THR, dstVal = minVal; lowThr < srcVal highThr, dstVal = srcVal; srcVal > highThr, dstVal = maxVal;

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_S16\_CTRL\_S

## 4.2.32 IVE\_THRESH\_S16\_CTRL\_S

**【Description】**

Define the thresholding control parameter of 16 bit signed image.

**【Syntax】**

```
typedef struct IVE_THRESH_S16_ CTRL
{
    IVE_THRESH_S16_MODE_E enMode;

    CVI_S16 s16LowThr;

    CVI_S16 s16HightThr;

    IVE_8BIT_U un8MinVal;

    IVE_8BIT_U un8MidVal;
```

(continues on next page)



(continued from previous page)

```

    IVE_8BIT_U un8MaxVal;

} IVE_THRESH_S16_ CTRL_S;

```

**【Member】**

Member	Description
enMode	Thresholding operation mode.
s16LowThr	Low threshold.
s16HightThr	High threshold.
un8MinVal	Minimum.
un8MidVal	Median.
un8MaxVal	Maximum.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_S16\_MODE\_E

### 4.2.33 IVE\_THRESH\_U16\_MODE\_E

**【Description】**

Define the thresholding mode of 16 bit unsigned image.

**【Syntax】**

```

typedef struct IVE_THRESH_U16_ MODE_E
{

    IVE_THRESH_U16_MODE_U16_TO_U8_MIN_MID_MAX=0x0,

    IVE_THRESH_U16_MODE_U16_TO_U8_MIN_ORI_MAX=0x1,

    IVE_THRESH_U16_MODE_BUTT

} IVE_THRESH_U16_MODE_E;

```

**【Member】**

Member	Description
IVE_THRESH_U16_MODE_U16_TO_U8_MIN_MID_MAX	MinVal, MidVal, MaxVal, dstVal = minVal; lowThr < srcVal highThr, dstVal = midVal; srcVal > highThr, dstVal = maxVal;
IVE_THRESH_U16_MODE_U16_TO_U8_MIN_ORI_MAX	MinVal, OriLowThr, MaxVal, dstVal = minVal; lowThr < srcVal highThr, dstVal = srcVal; srcVal > highThr, dstVal = maxVal;

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_U16\_CTRL\_S

## 4.2.34 IVE\_THRESH\_U16\_CTRL\_S

**【Description】**

Define the thresholding control parameter of 16 bit unsigned image.

**【Syntax】**

```
typedef struct IVE_THRESH_U16_ CTRL_S
{
    IVE_THRESH_U16_MODE_E enMode;

    CVI_U16 u16LowThr;

    CVI_U16 u16HightThr;

    IVE_8BIT_U u8MinVal;

    IVE_8BIT_U u8MidVal;

    IVE_8BIT_U u8MaxVal;
} cviIVE_THRESH_U16_ CTRL_S;
```

**【Member】**

Member	Description
enMode	Thresholding operation mode.
u16LowThr	Low threshold.
u16HightThr	High threshold.
u8MinVal	Minimum.
u8MidVal	Median.
u8MaxVal	Maximum.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_THRESH\_S16\_MODE\_E

### 4.2.35 IVE\_16BIT\_TO\_8BIT\_MODE\_E

**【Description】**

Define the conversion mode from 16 bit image data to 8 bit image data

**【Syntax】**

```
typedef enum cviIVE_16BIT_TO_8BIT_MODE_E
{
    IVE_16BIT_TO_8BIT_MODE_S16_TO_S8=0x0,
    IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_ABS=0x1,
    IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_BIAS=0x2,
    IVE_16BIT_TO_8BIT_MODE_S16_TO_U8=0x3,
    IVE_16BIT_TO_8BIT_MODE_BUTT
} IVE_16BIT_TO_8BIT_MODE_E;
```

**【Member】**

Member	Description
IVE_16BIT_TO_8BIT_MODE_S16_TO_S8	Linear transformation from S16 data to S8 data.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_ABS	Linear transformation from S16 data to S8 data, take the absolute value to get S8 data.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_BIAS	S16 data is linearly transformed to S8 data and truncated to U8 data after translation.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8	U16 data is linearly transformed to U8 data.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_16BIT\_TO\_8BIT\_CTRL\_S

### 4.2.36 IVE\_16BIT\_TO\_8BIT\_CTRL\_S

**【Description】**

Define the conversion control Parameter from 16 bit image data to 8 bit image data

**【Syntax】**

```
typedef struct cviIVE_16BIT_TO_8BIT_CTRL_S
{
    IVE_16BIT_TO_8BIT_MODE_E enMode;

    CVI_U16 u16Denominator;

    CVI_U8 u8Numerator;

    CVI_S8 s8Bias;
} IVE_16BIT_TO_8BIT_CTRL_S;
```

**【Member】**

Member	Description
enMode	The conversion mode from 16 bit data to 8 bit data.
u16Denominator	Denominator in linear transformation. Value range: [Max{1,u8Numerator}, 65535]
u8Numerator	Numerator in linear transformation. Value range: [0,255]。
s8Bias	Translation term in linear transformation. Value range: [- 128,127].

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_16BIT\_TO\_8BIT\_MODE\_E

### 4.2.37 IVE\_ORD\_STAT\_FILTER\_MODE\_E

**【Description】**

Define the order statistics filtering mode.

**【Syntax】**

```
typedef enum IVE_ORD_STAT_FILTER_MODE
{
    IVE_ORD_STAT_FILTER_MODE_MEDIAN = 0x0,
    IVE_ORD_STAT_FILTER_MODE_MIN = 0x1,
    IVE_ORD_STAT_FILTER_MODE_MAX = 0x2,
    IVE_ORD_STAT_FILTER_MODE_BUTT
} IVE_ORD_STAT_FILTER_MODE_E;
```

**【Member】**

Member	Description
IVE_ORD_STAT_FILTER_MODE_MEDIAN	The median filtering.
IVE_ORD_STAT_FILTER_MODE_MIN	The minimum filtering which is equivalent to the erosion of gray image.
IVE_ORD_STAT_FILTER_MODE_MAX	The maximum filtering which is equivalent to the dilation of gray image.

**【Note】**

None.

**【Related Data Type and Interface】**

ORD\_STAT\_FILTER\_CTRL\_S

### 4.2.38 IVE\_ORD\_STAT\_FILTER\_CTRL\_S

**【Description】**

Define the order statistics filter control parameter.

**【Syntax】**

```
typedef struct cviIVE_ORD_STAT_FILTER_CTRL_S
{
```

(continues on next page)

(continued from previous page)

```
IVE_ORD_STAT_FILTER _MODE_E enMode;  
}  
IVE_ORD_STAT_FILTER _CTRL_S;
```

**【Member】**

Member	Description
enMode	Order statistics filtering mode

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_ORD\_STAT\_FILTER \_MODE\_E

## 4.2.39 IVE\_MAP\_MODE\_E

**【Description】**

The MAP mode.

**【Syntax】**

```
typedef enum _IVE_MAP_CODE_E  
{  
  
    IVE_MAP_MODE_U8 = 0x0;  
  
    IVE_MAP_MODE_S16 = 0x1;  
  
    IVE_MAP_MODE_U16 = 0x2;  
  
} IVE_MAP_CODE_E;
```

**【Member】**

Member	Description
IVE_MAP_MODE_U8	U8C1 to U8C1Mapping
IVE_MAP_MODE_S16	U8C1 to U16C1Mapping
IVE_MAP_MODE_U16	U8C1 to S16C1Mapping

**【Note】**

None.

**【Related Data Type and Interface】**

None.

## 4.2.40 IVE\_ADD\_CTRL\_S

**【Description】**

Define the weighted addition control parameters for two images.

**【Syntax】**

```
typedef struct IVE_ADD_CTRL_S
{
    CVI_U0Q16 u0q16X;

    CVI_U0Q16 u0q16Y;
} IVE_ADD_CTRL_S;
```

**【Member】**

Member	Description
u0q16X	Weighted addition coefficient “X” in “XA + Yb”
u0q16Y	Weighted addition coefficient “Y” in “XA + Yb”

**【Note】**

None.

**【Related Data Type and Interface】**

None.

## 4.2.41 IVE\_NCC\_DST\_MEM\_S

**【Description】**

Define the output memory information of NCC.

**【Syntax】**

```
typedef struct cviIVE_NCC_DST_MEM_S
{
    CVI_U64 u64Numerator;

    CVI_U64 u64QuadSum1;

    CVI_U64 u64QuadSum2;
```

(continues on next page)

(continued from previous page)

```

    CVI_U8 u8Reserved[8];

} IVE_NCC_DST_MEM_S;

```

**【Member】**

Member	Description
u64Numerator	The numerator of the NCC formula $\sum_{i=1}^w \sum_{j=1}^h (I_{src1}(i, j) * I_{src2}(i, j))$
u64QuadSum1	The denominator of the NCC formula-Inner part of radical $\sum_{i=1}^w \sum_{j=1}^h (I_{src2}^2(i, j))$
u64QuadSum2	The denominator of the NCC formula-Inner part of radical $\sum_{i=1}^w \sum_{j=1}^h (I_{src1}^2(i, j)) \circ$
u8Reserved	The reserved field

**【Note】**

The calculation formula refers to **【Note】** in CVI\_IVE\_NCC.

**【Related Data Type and Interface】**

None.

## 4.2.42 IVE\_GMM\_CTRL\_S

**【Description】**

Define the control parameter of the GMM.

**【Syntax】**

```

typedef struct _IVE_GMM_CTRL_S {

    CVI_U22Q10 u22q10NoiseVar;

    CVI_U22Q10 u22q10MaxVar;

    CVI_U22Q10 u22q10MinVar;

```

(continues on next page)



(continued from previous page)

```
CVI_U0Q16 u0q16LearnRate;  
  
CVI_U0Q16 u0q16BgRatio;  
  
CVI_U8Q8 u8q8VarThr;  
  
CVI_U0Q16 u0q16InitWeight;  
  
CVI_U8 u8ModelNum;  
} IVE_GMM_CTRL_S;
```

**【Member】**

Member	Description
u22q10NoiseVar	initial noise variance Value range: [0x1, 0xFFFFFFFF]
u22q10MaxVar	The maximum variance of the mode Value range: [0x1, 0xFFFFFFFF]
u22q10MinVar	The minimum variance of the mode Value range: [1, u22q10MaxVar]
u0q16LearnRate	Learning rate Value range: [1, 65535]
u0q16BgRatio	The background ratio threshold Value range: [1, 65535]
u8q8VarThr	The threshold value of the variance Value range: [1, 65535]
u0q16InitWeight	The initial weight Value range: [1, 65535]
u8ModelNum	Several Gaussian models Value range: {3, 5}

**【Note】**

None.

**【Related Data Type and Interface】**

None.

### 4.2.43 IVE\_LBP\_CMP\_MODE\_E

**【Description】**

Define the comparison mode of LBP calculation.

**【Syntax】**

```
typedef enum cviIVE_LBP_CMP_MODE_E
{
    IVE_LBP_CMP_MODE_NORMAL = 0x0,
    IVE_LBP_CMP_MODE_ABS = 0x1,
    IVE_LBP_CMP_MODE_BUTT
} IVE_LBP_CMP_MODE_E;
```

**【Member】**

Member	Description
IVE_LBP_CMP_MODE_NORMAL	LBP simple comparison mode
IVE_LBP_CMP_MODE_ABS	LBP absolute value comparison mode

**【Note】**

The calculation formula refers to **【Note】** in CVI\_IVE\_LBP.

**【Related Data Type and Interface】**

IVE\_LBP\_CTRL\_S.

### 4.2.44 IVE\_LBP\_CTRL\_S

**【Description】**

Parameter of LBP texture.

**【Syntax】**

```
typedef struct cviIVE_LBP_CTRL_S
{
    IVE_LBP_CMP_MODE_E enMode;
    IVE_8BIT_U un8BitThr;
}IVE_LBP_CTRL_S;
```

**【Member】**

Member	Description
enMode	LBP comparison mode
un8BitThr	LBP comparison threshold. Value range is [- 128,127] in IVE_LBP_CMP_MODE_NORMAL; Value range is [0,255] in IVE_LBP_CMP_MODE_ABS

**【Note】**

The calculation formula refers to **【Note】** in CVI\_IVE\_LBP.

**【Related Data Type and Interface】**

IVE\_LBP\_CMP\_MODE\_E

IVE\_8BIT\_U

## 4.2.45 IVE\_NORM\_GRAD\_OUT\_CTRL\_E

**【Description】**

Define the task of calculating normalized gradient information and output the control enumeration type.

**【Syntax】**

```
typedef enum cviIVE_NORM_GRAD_OUT_CTRL_E
{
    IVE_NORM_GRAD_OUT_CTRL_HOR_AND_VER = 0x0,
    IVE_NORM_GRAD_OUT_CTRL_HOR = 0x1,
    IVE_NORM_GRAD_OUT_CTRL_VER = 0x2,
    IVE_NORM_GRAD_OUT_CTRL_COMBINE = 0x3,
    IVE_NORM_GRAD_OUT_CTRL_BUTT
} IVE_NORM_GRAD_CTRL_E;
```

**【Member】**

Member	Description
IVE_NORM_GRAD_OUT_CTRL_HOR_AND_VER	Output the H and V component images with gradient information simultaneously.
IVE_NORM_GRAD_OUT_CTRL_HOR	Only output the H component image with gradient information.
IVE_NORM_GRAD_OUT_CTRL_VER	Only output the V component image with gradient information.
IVE_NORM_GRAD_OUT_CTRL_COMBINE	The outputted gradient information is stored in package.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_NORM\_GRAD\_OUT\_CTRL\_S

## 4.2.46 IVE\_NORM\_GRAD\_CTRL\_S

**【Description】**

Define the control parameter of calculating normalized gradient information.

**【Syntax】**

```
typedef struct IVE_NORM_GRAD_CTRL {
    IVE_NORM_GRAD_OUT_CTRL_E enOutCtrl;

    IVE_MAG_DIST_E enDistCtrl;

    IVE_ITC_TYPE_E enITCType;

    CVI_U8 u8MaskSize;
} IVE_NORM_GRAD_CTRL_S;
```

**【Member】**

Member	Description
enOutCtrl	Output format
enDistCtrl	Distance calculation method
enITCType	Whether to do normalization.
u8MaskSize	Mask size

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_ITC\_CTRL\_S

IVE\_NORM\_GRAD\_OUT\_CTRL\_E

## 4.2.47 IVE\_SAD\_MODE\_E

### 【Description】

Define SAD calculation mode.

### 【Syntax】

```
typedef enum cviIVE_SAD_MODE_E
{
    IVE_SAD_MODE_MB_4x4 = 0x0,
    IVE_SAD_MODE_MB_8x8 = 0x1,
    IVE_SAD_MODE_MB_16x16 = 0x2,
    IVE_NORM_GRAD_OUT_CTRL_BUTT
} IVE_SAD_MODE_E;
```

### 【Member】

Member	Description
IVE_SAD_MODE_MB_4x4	SAD is calculated by 4x4 pixel block.
IVE_SAD_MODE_MB_8x8	SAD is calculated as a block of 8x8 pixels.
IVE_SAD_MODE_MB_16x16	SAD is calculated by 16x16 pixel block.

### 【Note】

None.

### 【Related Data Type and Interface】

IVE\_SAD\_CTRL\_S

## 4.2.48 IVE\_SAD\_OUT\_CTRL\_E

### 【Description】

Define SAD output mode.

### 【Syntax】

```
typedef enum cviIVE_SAD_OUT_CTRL_E
{
    IVE_SAD_OUT_CTRL_16BIT_BOTH = 0x0,
    IVE_SAD_OUT_CTRL_8BIT_BOTH = 0x1,
    IVE_SAD_OUT_CTRL_16BIT_SAD = 0x2,
    IVE_SAD_OUT_CTRL_8BIT_SAD = 0x3,
    IVE_SAD_OUT_CTRL_THRESH = 0x4,
    IVE_SAD_OUT_CTRL_BUTT
} IVE_SAD_OUT_CTRL_E;
```

【Member】

Member	Description
IVE_SAD_OUT_CTRL_16BIT_BOTH	16 bit SAD image and thresholding image output mode.
IVE_SAD_OUT_CTRL_8BIT_BOTH	8 bit SAD image and thresholding image output mode.
IVE_SAD_OUT_CTRL_16BIT_SAD	16 bit SAD image output mode.
IVE_SAD_OUT_CTRL_8BIT_SAD	8 bit SAD image output mode.
IVE_SAD_OUT_CTRL_THRESH	Thresholding image output mode.

【Note】

None.

【Related Data Type and Interface】

IVE\_SAD\_CTRL\_S

4.2.49 IVE\_SAD\_CTRL\_S

【Description】

Define SAD control Parameter

【Syntax】

```
typedef struct cviIVE_SAD_CTRL_S
{
    IVE_SAD_MODE_E enMode;
```

(continues on next page)

(continued from previous page)

```

IVE_SAD_OUT_CTRL_E enOutCtrl;

CVI_U16 u16Thr;

CVI_U8 u8MinVal;

CVI_U8 u8MaxVal;

} IVE_SAD_CTRL_S;

```

**【Member】**

Member	Description
enMode	SAD calculating mode.
enOutCtrl	SAD output control mode.
u16Thr	<p>The threshold value of thresholding the SAD image.</p> <p>The value range depends on enMode:</p> <p>1、IVE_SAD_OUT_CTRL_8BIT_BOTH, value [0, 255]</p> <p>2、IVE_SAD_OUT_CTRL_16BIT_BOTH 和 IVE_SAD_OUT_CTRL_THRESH, value [0, 65535]</p>
u8MinVal	The value when the thresholding value is less than u16Thr.
u8MaxVal	The value when the thresholding value exceeds u16Thr.

**【Note】**

None.

**【Related Data Type and Interface】**

IVE\_SAD\_MODE\_E

IVE\_SAD\_OUT\_CTRL\_E



## 4.2.50 IVE\_HOG\_CTRL\_S

### 【Description】

Define and calculate HOG (histogram of oriented gradient) feature control Parameter.

### 【Syntax】

```
typedef struct IVE_HOG_CTRL {  
  
    CVI_U8 u8BinSize;  
  
    CVI_U32 u32CellSize;  
  
    CVI_U16 u16BlkSizeInCell;  
  
    CVI_U16 u16BlkStepX;  
  
    CVI_U16 u16BlkStepY;  
  
} IVE_HOG_CTRL_S;
```

### 【Member】

Member	Description
u8BinSize	Number of histogram bin per cell
u32CellSize	Cell size
u16BlkSizeInCell	Block size contained in a cell
u16BlkStepX	Stride x
u16BlkStepY	Stride y

### 【Note】

None.

### 【Related Data Type and Interface】

None.

## 4.2.51 IVE\_GRAD\_FG\_CTRL\_S

### 【Description】

Define the Gradfg control parameter.

### 【Syntax】

```
typedef struct _IVE_GRAD_FG_CTRL_S {  
  
    IVE_GRAD_FG_MODE_E enMode;
```

(continues on next page)

(continued from previous page)

```

    CVI_U16 u16EdwFactor;

    CVI_U8 u8Cr1CoefThr;

    CVI_U8 u8MagCr1Thr;

    CVI_U8 u8MinMagDiff;

    CVI_U8 u8NoiseVal;

    CVI_U8 u8EdwDark;

} IVE_GRAD_FG_CTRL_S;

```

**【Member】**

Member	Description
enMode	Calculation mode
u16EdwFactor	Edge width adjustment factor (range: 500 to 2000; default: 1000)
u8Cr1CoefThr	Gradient vector correlation coefficient threshold (ranges: 50 to 100; default: 80)
u8MagCr1Thr	Gradient amplitude threshold (range: 0 to 20; default: 4)
u8MinMagDiff	Gradient magnitude difference threshold (range: 2 to 8; default: 2)
u8NoiseVal	Gradient amplitude noise threshold (range: 1 to 8; default: 1)
u8EdwDark	Black pixels enable flag (range: 0 (no), 1 (yes); default: 1)

**【Note】**

无。

**【Related Data Type and Interface】**

IVE\_GRAD\_FG\_MODE\_E

## 4.2.52 IVE\_GRAD\_FG\_MODE\_E

**【Description】**

Define Gradfg Calculation mode

**【Syntax】**

```
typedef enum _IVE_GRAD_FG_MODE_E {
```

(continues on next page)

(continued from previous page)

```

IVE_GRAD_FG_MODE_USE_CUR_GRAD = 0x0,

IVE_GRAD_FG_MODE_FIND_MIN_GRAD = 0x1,

IVE_GRAD_FG_MODE_BUTT
} IVE_GRAD_FG_MODE_E;

```

**【Member】**

Member	Description
IVE_GRAD_FG_MODE_USE_CUR_GRAD	Current position gradient calculation mode
IVE_GRAD_FG_MODE_FIND_MIN_GRAD	Peripheral minimum gradient calculation mode.

**【Note】**

None.

**【Related Data Type and Interface】**

None.

### 4.2.53 IVE\_16BIT\_TO\_8BIT\_MODE\_E

**【Description】**

Define the conversion mode from 16 bit image data to 8 bit image data

**【Syntax】**

```

typedef struct cviIVE_16BIT_TO_8BIT_CTRL_S
{
    IVE_16BIT_TO_8BIT_MODE_S16_TO_S8 = 0x0,

    IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_ABS = 0x1,

    IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_BIAS = 0x2,

    IVE_16BIT_TO_8BIT_MODE_U16_TO_U8 = 0x3,

    IVE_16BIT_TO_8BIT_MODE_BUTT
}IVE_16BIT_TO_8BIT_MODE_E;

```

**【Member】**

Member	Description
IVE_16BIT_TO_8BIT_MODE_S16_TO_S8	Linear transformation from S16 data to S8 data.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_ABS	After linear transformation from S16 data to S8 data, take the absolute value to get S8 data.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8_BIAS	S16 data is linearly transformed to S8 data and truncated to U8 data after translation.
IVE_16BIT_TO_8BIT_MODE_S16_TO_U8	U16 data is linearly transformed to U8 data.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_16BIT\_TO\_8BIT\_CTRL\_S

## 4.2.54 IVE\_16BIT\_TO\_8BIT\_CTRL\_S

**【Description】**

Define the conversion control Parameter from 16 bit image data to 8 bit image data

**【Syntax】**

```
typedef struct cviIVE_16BIT_TO_8BIT_CTRL_S
{
    IVE_16BIT_TO_8BIT_MODE_E enMode;

    CVI_U16 u16Denominator;

    CVI_U8 u8Numerator;

    CVI_S8 s8Bias;
}IVE_16BIT_TO_8BIT_CTRL_S;
```

**【Member】**

Member	Description
enMode	The conversion mode from 16 bit data to 8 bit data.
u16Denominator	Denominator in linear transformation. Value range: [Max {1, u8Numerator}, 65535]
u8Numerator	Numerator in linear transformation. Value range: [0,255].
s8Bias	Translation term in linear transformation. Value range: [- 128,127].

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_16BIT\_TO\_8BIT\_MODE\_E

## 4.2.55 IVE\_IVE\_TYPE\_E

**【Description】**

The normalization parameters.

**【Syntax】**

```
typedef enum IVE_ITC_TYPE {  
  
    IVE_ITC_SATURATE = 0x0,  
  
    IVE_ITC_NORMALIZE = 0x1,  
  
} IVE_ITC_TYPE_E;
```

**【Member】**

Member	Description
IVE_ITC_SATURATE	saturation
IVE_ITC_NORMALIZE	normalization

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_ITC\_CTRL\_S
- IVE\_NORM\_GRAD\_CTRL\_S

## 4.2.56 IVE\_IVE\_CTRL\_S

**【Description】**

The image type conversion parameters.

**【Syntax】**

```
typedef struct IVE_ITC_CTRL {  
  
    IVE_ITC_TYPE_E enType;  
  
} IVE_ITC_CTRL_S;
```

**【Member】**

Member	Description
enType	The normalization parameter.

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_ITC\_TYPE\_E

## 4.2.57 IVE\_BLOCK\_CTRL\_S

**【Description】**

IVE\_BLOCK control Parameter.

**【Syntax】**

```
typedef struct IVE_BLOCK_CTRL {  
  
    CVI_FLOAT f32BinSize;  
  
    CVI_U32 u32CellSize;  
  
} IVE_BLOCK_CTRL_S;
```

;

**【Member】**

Member	Description
f32Scale	After cell averaging, divide by scale value.
u32CellSize	Cell size

**【Note】**

None.

**【Related Data Type and Interface】**

- IVE\_ITC\_TYPE\_E

# 5 Tips Description

---

## 5.1 The additional Buffer

Currently, only UINT8/INT8/BF16 operations are supported. Any functionality beyond the UINT8 value range is implemented using BF16, which may result in slower performance and require additional buffer space as a temporary storage area.



# 6 FAQ

---

## 6.1 The Use of The Cache

The timing of memory usage in cache is determined by the algorithm software' s usage of the memory.

Since IVE directly reads DDR memory data, if the memory used has a cache, the cache must be constantly flushed to ensure data consistency.

Therefore, if there is no frequent RISC-V operations, it is recommended to use memory without a cache;

otherwise, it is recommended to use memory with a cache.

## 6.2 The config of the blnstant parameter

Introduction of the last parameter of each algorithm function in IVE: setting “True” uses busy waiting to wait for interrupt response, while setting “False” will move the program out of the RISC-V and wait for the interrupt signal to notify the IVE interrupt program to run.