Classification status: Top secret ( ) Secret ( ) Internal ( ) Public ( √ )

Rockchip Intelligent Video Engine

(Technology Department, Graphics Computing Platform Center)

|  |  |  |
| --- | --- | --- |
| File Status:  [ ] Modifying  [ √ ] Officially released | Current version: | V1.2 |
| do Author: | Chen Cheng |
| Compilation Editor: | Liu Wenjun |
| Review nuclear: | Xiong Wei |
| Completion Date: | 2022-07- 07 |

Rockchips​ Semiconductor Co. , Ltd ( All rights reserved . Reproduction is prohibited .)

Disclaimer

This document is provided "as is." Rockchip Electronics Co., Ltd. ("Rockchip") makes no representations or warranties, express or implied, as to the accuracy, reliability, completeness, merchantability, fitness for a particular purpose, or non-infringement of any statements, information, or content herein. This document is for reference only.

Due to product upgrades or other reasons, this document may be updated or revised from time to time without notice.

Trademark Notice

"Rockchip," "Rockchip Micro," and "Rockchip" are registered trademarks of Rockchip and are owned by Rockchip. All other registered trademarks or trademarks mentioned in this document are the property of their respective owners.

Copyright © 2022 Rockchip Electronics Co., Ltd.

Exceeding the scope of fair use, any entity or individual may not excerpt or reproduce the contents of this document, in whole or in part, or disseminate them in any form without the written permission of Rockchip.

Rockchip Electronics Co., Ltd.

Rockchip Electronics Co., Ltd.

Address: Software Park A, Tongpan Road, Fuzhou, Fujian District 18 Website : [www.rock-chips.com](http://www.rock-chips.com/)

Customer Service Hotline: + 86-4007-700-590

Customer Service Fax: +86-591-83951833

Customer Service Email: [fae@rock-chips.com](mailto:fae@rock-chips.com)

## Preface

##### Target audience

* Software Development Engineer
* Technical Support Engineer

##### Applicable platforms

* RV1103
* RV1106

##### Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version Information | Version Notes | date | author |
| **V1.0** | Initial version | 2022.04.06 | Chen Cheng |
| **V1.1** | Add schematic diagram and improve interface description | 2022.04.26 | Chen Cheng |
| **V1.2** | Reorganize the directory structure, add affine transformation and create image pyramid interface instructions | 2022.07.07 | Chen Cheng |

# Table of contents

[Preface](#_bookmark0)  [3](#_bookmark0)

[Document Applicable Objects](#_bookmark1)  [3](#_bookmark1)

[Applicable platform](#_bookmark2)  [3](#_bookmark2)

[Version Record](#_bookmark3)  [3](#_bookmark3)

[Directory](#_bookmark4)  [4](#_bookmark4)

1. [Overview](#_bookmark5)  [7](#_bookmark5)
2. [Getting Started Guide](#_bookmark6)  [10](#_bookmark6)
   1. [Environmental Description](#_bookmark7)  [10](#_bookmark7)
      1. [Directory structure description](#_bookmark8)  [10](#_bookmark8)
      2. [Input and output buffer](#_bookmark9)  [11](#_bookmark9)
   2. [Basic Concepts](#_bookmark10)  [11](#_bookmark10)
   3. [Reference Example](#_bookmark11)  [13](#_bookmark11)
3. [Proc Debug information](#_bookmark12)  [16](#_bookmark12)
   1. [Overview](#_bookmark13)  [16](#_bookmark13)
   2. [Proc Information Note](#_bookmark14)  [16](#_bookmark14)
   3. [Error code](#_bookmark15)  [18](#_bookmark15)
4. [Image format diagram](#_bookmark17)  [19](#_bookmark17)

[YCbCr 4:0:0 format](#_bookmark18)  [19](#_bookmark18)

[YUV420SP format](#_bookmark19)  [19](#_bookmark19)

[YUV420P format](#_bookmark20)  [20](#_bookmark20)

[YUV422SP format](#_bookmark21)  [20](#_bookmark21)

[YUV422P format](#_bookmark22)  [21](#_bookmark22)

[U8C1 format](#_bookmark23)  [21](#_bookmark23)

[U8C3 planar format](#_bookmark24)  [22](#_bookmark24)

[U8C3 package format](#_bookmark25)  [22](#_bookmark25)

1. [API Reference](#_bookmark26)  [23](#_bookmark26)

[RK\_MPI\_IVE\_Init](#_bookmark27)  [25](#_bookmark27)

[RK\_MPI\_IVE\_Deinit](#_bookmark28)  [25](#_bookmark28)

[RK\_MPI\_IVE\_CvtImageToData](#_bookmark29)  [26](#_bookmark29)

[RK\_MPI\_IVE\_CvtDataToImage](#_bookmark30)  [27](#_bookmark30)

[RK\_MPI\_IVE\_CvtImageToMemInfo](#_bookmark31)  [28](#_bookmark31)

[RK\_MPI\_IVE\_DMA](#_bookmark32)  [29](#_bookmark32)

[RK\_MPI\_IVE\_Filter](#_bookmark33)  [31](#_bookmark33)

[RK\_MPI\_IVE\_CSC](#_bookmark34)  [33](#_bookmark34)

[RK\_MPI\_IVE\_Sobel](#_bookmark35)  [35](#_bookmark35)

[RK\_MPI\_IVE\_MagAndAng](#_bookmark36)  [37](#_bookmark36)

[RK\_MPI\_IVE\_Dilate](#_bookmark37)  [40](#_bookmark37)

[RK\_MPI\_IVE\_Erode](#_bookmark38)  [42](#_bookmark38)

[RK\_MPI\_IVE\_Add](#_bookmark40)  [43](#_bookmark40)

[RK\_MPI\_IVE\_And](#_bookmark42)  [45](#_bookmark42)

[RK\_MPI\_IVE\_Sub](#_bookmark44)  [46](#_bookmark44)

[RK\_MPI\_IVE\_Or](#_bookmark46)  [48](#_bookmark46)

[RK\_MPI\_IVE\_Xor](#_bookmark48)  [49](#_bookmark48)

[RK\_MPI\_IVE\_Integ](#_bookmark50)  [51](#_bookmark50)

[RK\_MPI\_IVE\_Hist](#_bookmark52)  [53](#_bookmark52)

[RK\_MPI\_IVE\_Thresh](#_bookmark53)  [55](#_bookmark53)

[RK\_MPI\_IVE\_Thresh\_u16](#_bookmark54)  [57](#_bookmark54)

[RK\_MPI\_IVE\_Thresh\_s16](#_bookmark55)  [58](#_bookmark55)

[RK\_MPI\_IVE\_16bitto8bit](#_bookmark57)  [60](#_bookmark57)

[RK\_MPI\_IVE\_8bitto8bit](#_bookmark58)  [61](#_bookmark58)

[RK\_MPI\_IVE\_OrdStatFilter](#_bookmark60)  [63](#_bookmark60)

[RK\_MPI\_IVE\_Map](#_bookmark61)  [65](#_bookmark61)

[RK\_MPI\_IVE\_EqualizeHist](#_bookmark62)  [67](#_bookmark62)

[RK\_MPI\_IVE\_Ncc](#_bookmark63)  [69](#_bookmark63)

[RK\_MPI\_IVE\_CCL](#_bookmark65)  [70](#_bookmark65)

[RK\_MPI\_IVE\_Gmm](#_bookmark66)  [72](#_bookmark66)

[RK\_MPI\_IVE\_Gmm2](#_bookmark67)  [74](#_bookmark67)

[RK\_MPI\_IVE\_CannyEdge](#_bookmark69)  [78](#_bookmark69)

[RK\_MPI\_IVE\_LBP](#_bookmark70)  [80](#_bookmark70)

[RK\_MPI\_IVE\_NormGrad](#_bookmark71)  [82](#_bookmark71)

[RK\_MPI\_IVE\_LKOpticalFlowPyr](#_bookmark72)  [83](#_bookmark72)

[RK\_MPI\_IVE\_LKOpticalFlow](#_bookmark73)  [85](#_bookmark73)

[RK\_MPI\_IVE\_STCandiCorner](#_bookmark74)  [87](#_bookmark74)

[RK\_MPI\_IVE\_STCorner](#_bookmark75)  [89](#_bookmark75)

[RK\_MPI\_IVE\_MatchBgModel](#_bookmark76)  [91](#_bookmark76)

[RK\_MPI\_IVE\_UpdateBgModel](#_bookmark77)  [93](#_bookmark77)

[RK\_MPI\_IVE\_SAD](#_bookmark78)  [96](#_bookmark78)

[RK\_MPI\_IVE\_Warp\_Affine\_Init](#_bookmark79)  [98](#_bookmark79)

[RK\_MPI\_IVE\_Warp\_Affine](#_bookmark80)  [99](#_bookmark80)

[RK\_MPI\_IVE\_Pyramid\_GetSize](#_bookmark81)  [100](#_bookmark81)

[RK\_MPI\_IVE\_Pyramid\_Create](#_bookmark82)  [101](#_bookmark82)

[RK\_MPI\_IVE\_Query](#_bookmark83)  [103](#_bookmark83)

1. [Data Types and Structures](#_bookmark84)  [105](#_bookmark84)

[IVE\_IMAGE\_S](#_bookmark85)  [105](#_bookmark85)

[IVE\_SRC\_IMAGE\_S](#_bookmark86)  [105](#_bookmark86)

[IVE\_DST\_IMAGE\_S](#_bookmark87)  [105](#_bookmark87)

[IVE\_DATA\_S](#_bookmark88)  [106](#_bookmark88)

[IVE\_SRC\_DATA\_S](#_bookmark89)  [106](#_bookmark89)

[IVE\_DST\_DATA\_S](#_bookmark90)  [106](#_bookmark90)

[IVE\_MEM\_INFO\_S](#_bookmark91)  [107](#_bookmark91)

[IVE\_SRC\_MEM\_INFO\_S](#_bookmark92)  [107](#_bookmark92)

[IVE\_DST\_MEM\_INFO\_S](#_bookmark93)  [107](#_bookmark93)

[IVE\_DMA\_CTRL\_S](#_bookmark94)  [108](#_bookmark94)

[IVE\_FILTER\_CTRL\_S](#_bookmark95)  [109](#_bookmark95)

[IVE\_CSC\_CTRL\_S](#_bookmark96)  [110](#_bookmark96)

[IVE\_SOBEL\_CTRL\_S](#_bookmark98)  [112](#_bookmark98)

[IVE\_MAG\_AND\_ANG\_CTRL\_S](#_bookmark99)  [113](#_bookmark99)

[IVE\_DILATE\_CTRL\_S](#_bookmark100)  [113](#_bookmark100)

[IVE\_ERODE\_CTRL\_S](#_bookmark101)  [114](#_bookmark101)

[IVE\_ADD\_CTRL\_S](#_bookmark102)  [114](#_bookmark102)

[IVE\_SUB\_CTRL\_S](#_bookmark104)  [115](#_bookmark104)

[IVE\_INTEG\_CTRL\_S](#_bookmark105)  [115](#_bookmark105)

[IVE\_THRESH\_CTRL\_S](#_bookmark106)  [116](#_bookmark106)

[IVE\_8BIT\_TO\_8BIT\_CTRL\_S](#_bookmark107)  [118](#_bookmark107)

[IVE\_16BIT\_TO\_8BIT\_CTRL\_S](#_bookmark108)  [118](#_bookmark108)

[IVE\_ORD\_STAT\_FILTER\_CTRL\_S](#_bookmark109)  [120](#_bookmark109)

[IVE\_MAP\_CTRL\_S](#_bookmark110)  [121](#_bookmark110)

[IVE\_EQHIST\_CTRL\_S](#_bookmark111)  [121](#_bookmark111)

[IVE\_CCL\_CTRL\_S](#_bookmark112)  [122](#_bookmark112)

[IVE\_CANNY\_EDGE\_CTRL\_S](#_bookmark113)  [123](#_bookmark113)

[IVE\_LBP\_CTRL\_S](#_bookmark114)  [124](#_bookmark114)

[IVE\_GMM\_CTRL\_S](#_bookmark116)  [125](#_bookmark116)

[IVE\_GMM2\_CTRL\_S](#_bookmark117)  [127](#_bookmark117)

[IVE\_LK\_OPTICAL\_FLOW\_CTRL\_S](#_bookmark118)  [130](#_bookmark118)

[IVE\_LK\_OPTICAL\_FLOW\_PYR\_CTRL\_S](#_bookmark119)  [131](#_bookmark119)

[IVE\_ST\_CANDI\_CORNER\_CTRL\_S;](#_bookmark120)  [132](#_bookmark120)

[IVE\_ST\_CORNER\_CTRL\_S;](#_bookmark121)  [133](#_bookmark121)

[IVE\_MATCH\_BG\_MODEL\_CTRL\_S](#_bookmark122)  [134](#_bookmark122)

[IVE\_UPDATE\_BG\_MODEL\_CTRL\_S](#_bookmark123)  [135](#_bookmark123)

[IVE\_SAD\_CTRL\_S](#_bookmark124)  [136](#_bookmark124)

[IVE\_WARP\_AFFINE\_CTRL\_S](#_bookmark125)  [137](#_bookmark125)

[IVE\_PYRAMID\_CTRL\_S](#_bookmark126)  [138](#_bookmark126)

## Overview

RKIVE (Rockchip Intelligent Video Engine) is a hardware acceleration module for video image analysis in Rockchip media processing chips . It is used to accelerate video image analysis and reduce CPU The current hardware version integrates 22 The 3D POSIX ...

|  |  |  |
| --- | --- | --- |
| **Integral**  used as a quick and effective way of calculating the sum of pixel value | **Canny Edge**  uses a multi-stage algorithm to detect a wide range of edges | **Histogram Equalization**  processing of contrast adjustment using the image's histogram |
| **Threshold**  method of segmenting images, creating a binary images | **Filter**  Filter image with a 5x5 kernel | **Map**  map one set of pixel values to another |
| **Min Filter**  erodes shapes on the image | **Median Filter**  makes the target pixel luminosity equal to the mean value in the running window | **Max Filter**  extends object boundaries on the image |

|  |  |  |
| --- | --- | --- |
| **Erode**  removes pixels on object boundaries | **LBP**  labels the pixels of an image by thresholding the neighborhood of each pixel | **Cast**  Convert pixel value range to another |
| **Shi-tomasi corner**  finds N strongest corners in the image by Shi-Tomasi method | **Histogram**  acts as a graphical representation of the luminance distribution | **Sub- Abs**  subtract two images and calculate the absolute value |
| **Sub-Shift**  subtract two images and right shift 1 bit of the result | **Add**  combine two images with different weight | **And**  bitwise and of two images |
| **Or**  bitwise or of two images | **Xor**  bitwise xor of two images | **CCL**  Connected-component labeling , identify blobs of pixels in an image |

|  |  |  |
| --- | --- | --- |
| **SAD**  sum of absolute differences , a measure of the similarity between image blocks | **NCC**  normalized cross-correlation, can be used to determine how to register or align the images | **LK Optical Flow**  optical flow estimation |
| **GMM & BGM**  Background subtraction, distinguish foreground objects from the background | **Gradient Magnitude**  measure how strong the change in image intensity is | **Gradient Direction**  a directional change in image intensity |

## Getting Started

##### Environmental Description

###### Directory structure description

RKIVE The software includes the following parts, based on RV1103/RV1106 SDK For example:

media/ive/ive/

├── CMakeLists.txt

**├── include #RKIVE** Related header files

│ ├── rk\_comm\_ive.h

│ ├── rk\_ive.h

│ └── rk\_mpi\_ive.h

├── lib #RKIVE/RKIVS Related library files

│ ├── libivs.a #RKIVS Static library

│ ├── libivs.so #RKIVS Dynamic Library

│ ├── librve.a #RKIVE Static library

│ └── librve.so #RKIVE Dynamic Library

**├── sample #RKIVE** Calculate **the sample of the image histogram**

│ ├── main.c

│ ├── sample\_comm\_mem.h

│ ├── sample\_comm\_rve.c

│ ├── sample\_comm\_rve.h

│ └── sample\_rve\_mem.c

└── simulator

**#RKIVE simulator Windows-** based **visual studio** Simulation development environment

The simulator The file contains RKIVE Reference examples for all interface calls can be verified and developed based on the Windows Visual Studio environment, or ported to the SDK development board environment for operation.

RKIVE The path of the device driver in the development board file system is /oem/usr/ko/rve.ko . Check before running

/dev/rve Check whether the node exists to confirm whether the device driver is loaded. If automatic loading at startup is not configured, you can modify the script /oem/usr/ko/insmod\_ko.sh and add insmod rve.ko at the end of the script .

###### Input and output buffering

RKIVE The input [and](#_bookmark91) output buffer types required by the functional interface include [IVE\_MEM\_INFO\_S](#_bookmark91) , [IVE\_DATA\_S](#_bookmark88) , and [IVE\_DST\_IMAGE\_S](#_bookmark87) . On different platforms, memory is allocated based on the different buffer types and the corresponding buffer structures are filled. For example, in the RV1103/RV1106 SDK , RK\_MPI\_MMZ\_Alloc is called to allocate physically contiguous memory blocks, and then RK\_MPI\_MMZ\_Free is called to release the allocated memory. For more information, refer to sample\_comm\_rve.c in the sample code . The corresponding interface of file encapsulation is used to package input and output buffers:

* + - * SAMPLE\_COMM\_IVE\_CreateImage
      * SAMPLE\_COMM\_IVE\_CreateMemInfo
      * SAMPLE\_COMM\_IVE\_CreateData

After the operator is finished running, the MmzFree interface is called, the virtual address corresponding to the cache is passed in, and the corresponding cache is released . The specific implementation of MmzFree is to get the MB\_BLK pointer by the cache virtual address and call RK\_MPI\_MMZ\_Free Complete memory release.

##### Basic Concepts

* Line stride **or** virtual width

The memory space occupied by each row of an image or two-dimensional data. The row span is generally set to a value greater than or equal to the width to meet hardware alignment requirements .

* Address alignment

Some operators require the allocated input and output buffer first address to be 1 byte or 16 bytes Alignment, see API for details illustrate.

* **CPU cache**

When there is a CPU In the case of intervention, the cacheable When pre-processing or post-processing the input and output cache contents, you need to call RK\_MPI\_MMZ\_FlushCacheVaddrStart before processing and call RK\_MPI\_MMZ\_FlushCacheVaddrEnd after processing is completed. Refresh the cache . Prevent input and output data from being lost due to cache Error caused by not refreshing.

* One-dimensional data

One-dimensional data is a linear physical cache. The corresponding structure is [IVE\_MEM\_INFO\_S](#_bookmark91) , which contains information such as physical address, virtual address, and cache size.

* Two-dimensional data

Two-dimensional data is a physical cache of specific width, height and span information. The corresponding structure is

[IVE\_DATA\_S](#_bookmark88) .

* 2D images

A two-dimensional image is a physical buffer with a specific width, height, stride, and data channels. The corresponding structure is [IVE\_DST\_IMAGE\_S](#_bookmark87) . It can be used to represent image data with one or more channels. The width and height of each channel are consistent, and the stride, starting virtual address, and starting physical address are stored in corresponding arrays.

* Task **ID** ( **handle** )

The system assigns a task ID to each operator call . Is a constant greater than zero. If the returned task

ID Less than zero, indicating that the task creation failed.

* Task query ( **query** )

ID returned by the operator call into [RK\_MPI\_IVE\_Query](#_bookmark83) Interface to query whether the current node task is completed.

* Synchronous mode and asynchronous mode ( **bInstant** )

Each operator calls the API All with bInstant Parameters. When bInstant If set to RK\_TRUE , the current call will be blocked until the hardware completes the execution and returns the result. If set to RK\_FALSE , the current call returns immediately and forms a linked list with subsequent operators to execute in batch mode. This can reduce the number of interrupts, lower the CPU load , and improve performance.

##### Reference Examples

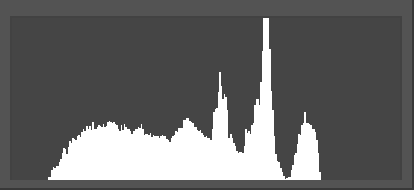


Figure 2-1​

Taking the statistical image histogram as a reference example, the sample code mainly includes the following parts:

* initialization

Call RK\_MPI\_IVE\_Init Initialize.

* Memory allocation

Call RK\_MPI\_MMZ\_Alloc to allocate memory for the input image and output histogram cache. The memory type requested is RK\_MMZ\_ALLOC\_CACHEABLE .

* run

Calling RK\_MPI\_IVE\_Hist Trigger the hardware to start executing the histogram statistics task.

* Query

Calling RK\_MPI\_IVE\_Query Query the running status of the histogram statistics task and wait for the task to complete.

* Post-processing

The CPU prints the histogram statistics of the hardware output to ddr . During CPU access, call RK\_MPI\_MMZ\_FlushCacheVaddrStart and RK\_MPI\_MMZ\_FlushCacheVaddrEnd Refresh the CPU cache to prevent data errors caused by cache refresh.

* Finish

Call RK\_MPI\_MMZ\_Free Release the previously allocated buffer and call RK\_MPI\_IVE\_Deinit Perform deinitialization.

**Sample Code:**

#include <rk\_mpi\_ive.h> #include <rk\_mpi\_mmz.h>

int main (void) { RK\_S32 s32Ret = 0;

bool bInstant = false ; bool bBlock = true ; bool bFinish = false ;

RK\_U32 u32Width = 16 ; RK\_U32 u32Height = 16 ; RK\_U32 u32Size = 0;

IVE\_HANDLE IveHandle = 0; IVE\_SRC\_IMAGE\_S stSrc = {0};

IVE\_DST\_MEM\_INFO\_S stHist = {0};

MB\_BLK stMB = NULL ;

RK\_U32\* pu32Hist = NULL ;

// initialize IVE context

s32Ret = RK\_MPI\_IVE\_Init (); if (s32Ret < 0 ) {

printf (stderr, "RK\_MPI\_IVE\_Init failed \n "); goto End;

}

// allocate memory for input image u32Size = u32Width \* u32Height;

s32Ret = RK\_MPI\_MMZ\_Alloc (&stMB, u32Size, RK\_MMZ\_ALLOC\_TYPE\_CMA); if (s32Ret == 0 ) {

stSrc . au64PhyAddr [ 0 ] = RK\_MPI\_MB\_Handle2PhysAddr (stMB); stSrc . au64VirAddr [ 0 ] = RK\_MPI\_MB\_Handle2VirAddr (stMB); stSrc . au32Stride [ 0 ] = u32Width;

stSrc.u32Width = u32Width ; stSrc . u32Height = u32Height;

} else {

printf (stderr, "Create input image mem failed \n "); goto End;

}

// allocate memory for histogram buffer

u32Size = IVE\_HIST\_NUM \* sizeof (RK\_U32);

s32Ret = RK\_MPI\_MMZ\_Alloc (&stMB, u32Size, RK\_MMZ\_ALLOC\_TYPE\_CMA);

if (s32Ret == 0 ) {

stHist . u64PhyAddr = RK\_MPI\_MB\_Handle2PhysAddr (stMB); stHist .u64VirAddr = RK\_MPI\_MB\_Handle2VirAddr (stMB) ; stHist . u32Size = u32Size;

} else {

printf (stderr, "Create hist mem info failed \n "); goto End;

}

// run histogram

s32Ret = RK\_MPI\_IVE\_Hist (&IveHandle, & stSrc, &stHist, bInstant); if (s32Ret < 0 ) {

printf (stderr, "RK\_MPI\_IVE\_Hist failed \n "); goto End;

}

// wait until histogram finished

s32Ret = RK\_MPI\_IVE\_Query (IveHandle, &bFinish, bBlock); while (ERR\_IVE\_QUERY\_TIMEOUT == s32Ret) {

usleep ( 100 );

s32Ret = RK\_MPI\_IVE\_Query (IveHandle, &bFinish, bBlock);

}

if (s32Ret < 0 ) {

printf (stderr, "RK\_MPI\_IVE\_Query failed \n "); goto End;

}

// get histogram result

RK\_MPI\_MMZ\_FlushCacheVaddrStart ( stHist . u64VirAddr , stHist . u32Size , RK\_MMZ\_ALLOC\_TYPE\_CMA);

pu32Hist = (RK\_U32) stHist . u64VirAddr ; printf ("Histogram test, hist[0] = %d \n ", pu32Hist [ 0 ]);

RK\_MPI\_MMZ\_FlushCacheVaddrEnd ( stHist . u64VirAddr , stHist . u32Size , RK\_MMZ\_ALLOC\_TYPE\_CMA); End:

// free memory

if ( stSrc . au64VirAddr [ 0 ] != 0 ) {

stMB = RK\_MPI\_MB\_VirAddr2Handle (( void \*) stSrc . au64VirAddr [ 0 ]); RK\_MPI\_MMZ\_Free (stMB);

stSrc . au64VirAddr [ 0 ] = 0 ;

stSrc . au64PhyAddr [ 0 ] = 0 ;

}

if ( stHist . u64VirAddr != 0 ) {

stMB = RK\_MPI\_MB\_VirAddr2Handle (( void \*) stHist . u64VirAddr ); RK\_MPI\_MMZ\_Free (stMB);

stHist . u64VirAddr = 0 ;

stHist . u64PhyAddr = 0 ;

}

// free IVE context

s32Ret = RK\_MPI\_IVE\_Deinit (); if (s32Ret < 0 ) {

printf (stderr, "RK\_MPI\_IVE\_Deinit failed \n ");

}

return 0 ;

}

1. **Proc** Debug information

##### Overview

Debug information is stored in Linux proc​ The file system records hardware operation information in real time for problem location and performance analysis.

* File Node

**/proc/rve**

* Instructions

To turn on or off debug information output, use the following command:

**# echo mon > /proc/rve/debug [311610.143190] rve\_debugger: close monitor!**

**# echo mon > /proc/rve/debug [311611.686203] rve\_debugger: open monitor!**

* 1. **Proc** Information Description
* Debug information:
  + scheduler : hardware processor entity
  + pd\_ref ： power Reference count statistics
  + total\_int\_cnt : hardware interrupt count
  + rd\_bandwidth: Read DDR Data volume
  + wr\_bandwidth : write ddr Data volume
  + cycle\_cnt/s : cycles consumed per second number
* View the current task status information:

**# cat /proc/rve/scheduler\_status num of scheduler = 1**

**===================================**

**scheduler[0]: rve**

**---------------------------------- -**

**pd\_ref = 17**

**total\_int\_cnt = 17**

**rd\_bandwidth: 256 bytes/s wr\_bandwidth: 1024 bytes/s**

**cycle\_cnt/s: 685**

* Record the last **10** Status information of each task:

**# cat /proc/rve/load num of scheduler = 1**

**================= load ==================**

**scheduler[0]: rve load = 0**

**---------------- PID INFO -------------- -**

**[pid: 3958] hw\_time\_total = 0 us**

**last\_job\_rd\_bandwidth: 256 bytes/s**

**last\_job\_wr\_bandwidth: 1024 bytes/s**

**last\_job\_cycle\_cnt/s: 685**

##### [Error Code](#_bookmark16)

If the operation fails, please check the returned error code. The error code description is shown in the following table:

|  |  |  |
| --- | --- | --- |
| Error Code | Macro Definition | describe |
| **0x40** | **ERR\_IVE\_SYS\_TIMEOUT** | System timeout |
| **0x41** | **ERR\_IVE\_QUERY\_TIMEOUT** | Query timeout |
| **0x42** | **ERR\_IVE\_OPEN\_FILE** | Open file timeout |
| **0x43** | **ERR\_IVE\_READ\_FILE** | Reading file timeout |
| **0x44** | **ERR\_IVE\_WRITE\_FILE** | Timeout on writing file |
| **0x45** | **ERR\_IVE\_BUS\_ERR** | Bus Error |
| **0x46** | **ERR\_IVE\_ILLEGAL\_PARAMS** | Illegal parameters |
| **0x47** | **ERR\_IVE\_DEVICE\_ERROR** | Device driver failure |
| **0x48** | **ERR\_IVE\_NOT\_SUPPORT** | Unsupported Operation |
| **0x49** | **ERR\_IVE\_ILLEGAL\_STMEM** | Illegal auxiliary memory |

## Image format diagram

#### YCbCr 4:0:0 format

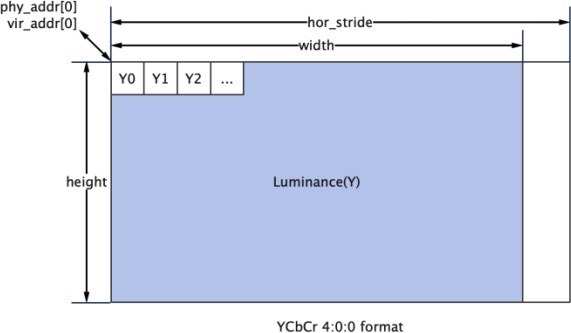
****

Figure 4-1​

#### YUV420SP format

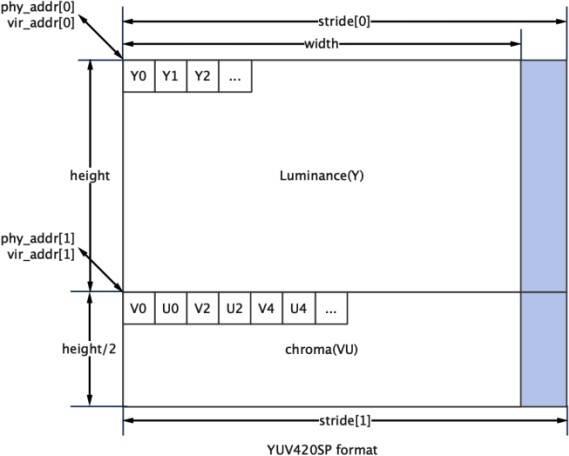
****

Figure 4-2​

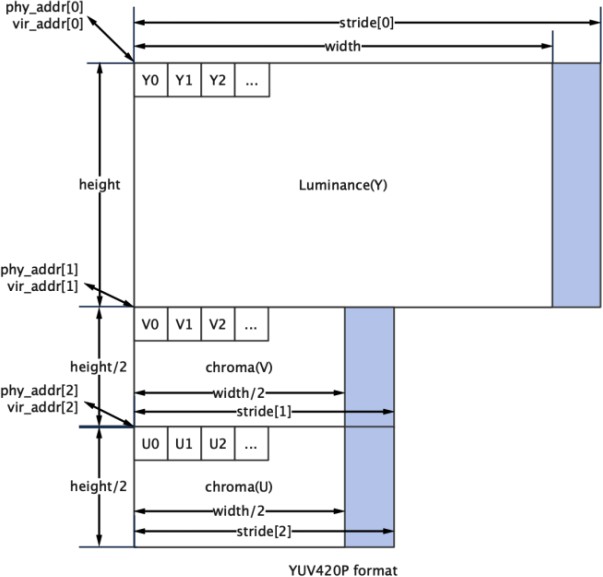


Figure 4-3​

#### YUV422SP format

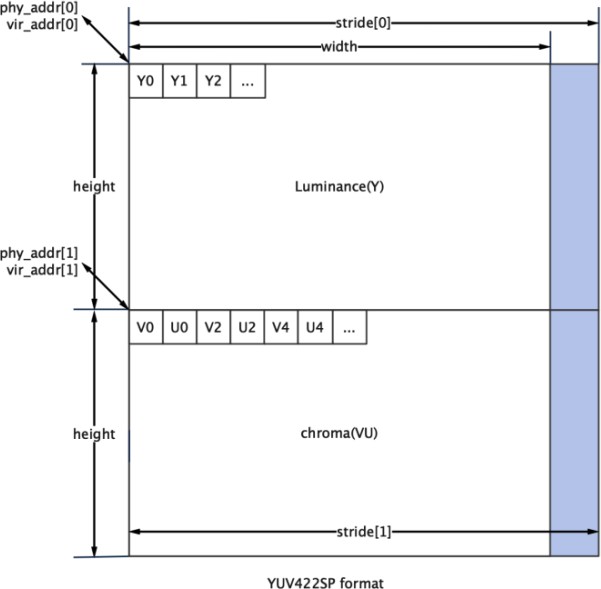
****

Figure 4- 4

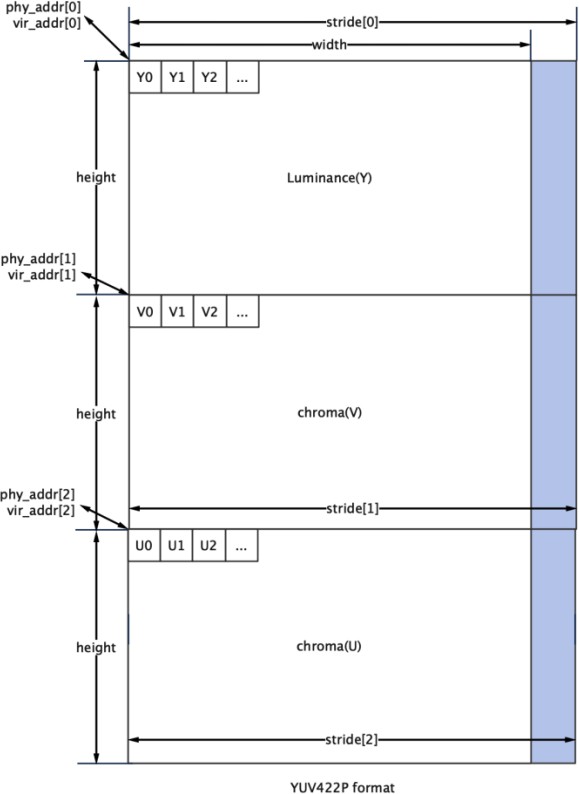


Figure 4-5​

#### U8C1 format

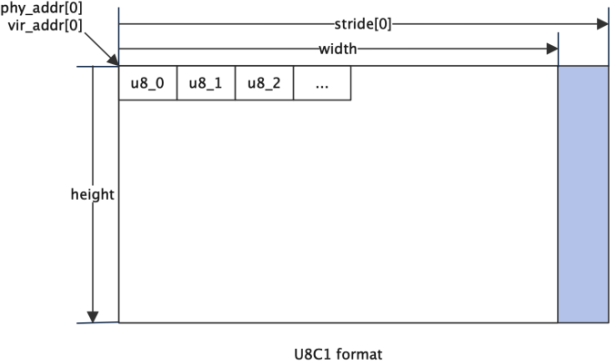
****

Figure 4-6​

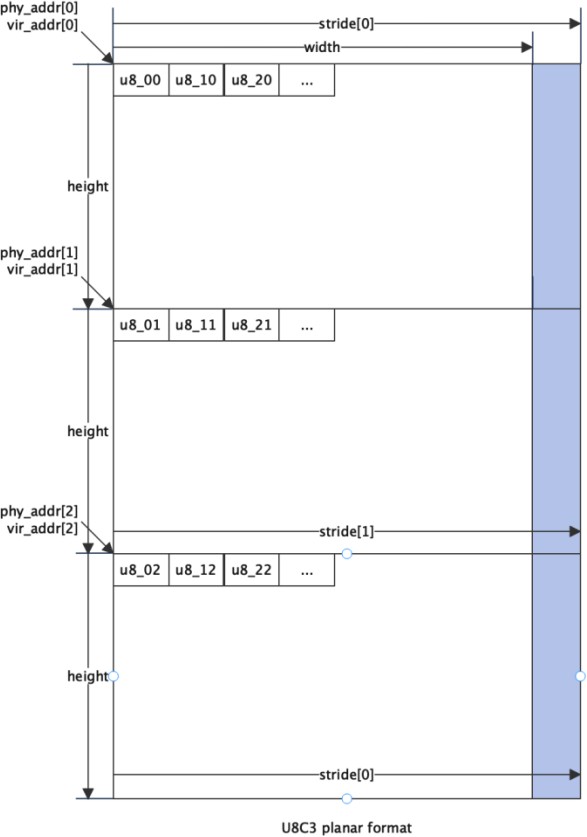


Figure 4-7​

#### U8C3 package format

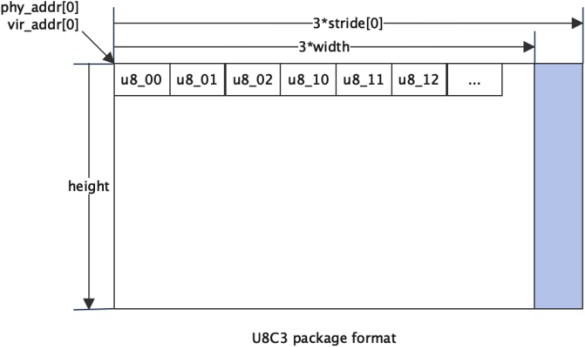
****

Figure 4-8​

1. **API** refer to

RKIVE Provides the following functional interfaces:

* [RK\_MPI\_IVE\_Init](#_bookmark27) : Complete IVE Context creation and necessary initialization work.
* [RK\_MPI\_IVE\_Deinit](#_bookmark28) : Complete IVE The context is destroyed.
* [RK\_MPI\_IVE\_CvtImageToData :](#_bookmark29) A single planar Image type is converted to type.
* [RK\_MPI\_IVE\_CvtDataToImage :](#_bookmark30) Converts a two-dimensional data type to a single planar Image type.
* [RK\_MPI\_IVE\_CvtImageToMemInfo :](#_bookmark31) A single planar Converts the image type to a one-dimensional data type.
* [RK\_MPI\_IVE\_DMA](#_bookmark32) : Direct memory access, supporting fast copy, interval copy, and memory filling.
* [RK\_MPI\_IVE\_Filter](#_bookmark33) : 5X5 Template filtering tasks can achieve different filtering by configuring different template coefficients.
* [RK\_MPI\_IVE\_CSC](#_bookmark34) : color space conversion, which can realize YUV , RGB , HSV Color space conversion.
* [RK\_MPI\_IVE\_Sobel](#_bookmark35) : 3X3 or 5X5 Template SOBEL-LIKE Gradient calculation.
* [RK\_MPI\_IVE\_MagAndAng :](#_bookmark36) Calculate the amplitude argument.
* [RK\_MPI\_IVE\_Dilate :](#_bookmark37) Image dilation, image morphological operation.
* [RK\_MPI\_IVE\_Erode :](#_bookmark39) Image erosion, image morphological operations.
* [RK\_MPI\_IVE\_Add :](#_bookmark40) Weighted addition operation of two grayscale images.
* [RK\_MPI\_IVE\_And :](#_bookmark41) Perform AND operation on two grayscale images.
* [RK\_MPI\_IVE\_Sub :](#_bookmark43) Subtract two grayscale images.
* [RK\_MPI\_IVE\_Or :](#_bookmark45) OR operation between two grayscale images.
* [RK\_MPI\_IVE\_Xor :](#_bookmark47) Perform the XOR operation on two grayscale images.
* [RK\_MPI\_IVE\_Integ :](#_bookmark49) Image integral map calculation.
* [RK\_MPI\_IVE\_Hist :](#_bookmark51) Image histogram calculation.
* [RK\_MPI\_IVE\_Thresh :](#_bookmark53) Image 8 Grayscale image binarization operation.
* [RK\_MPI\_IVE\_Thresh\_U](#_bookmark54) 16 : 16 bit data to 8 Bit data threshold binarization operation.
* [RK\_MPI\_IVE\_Thresh\_S16 :](#_bookmark56) 16 with sign bit bit data to 8 Threshold binarization operation for bit data.
* [RK\_MPI\_IVE\_16BitTo8Bit :](#_bookmark57) 16 bit data to 8 Linear conversion of bit data.
* [RK\_MPI\_IVE\_8BitTo8Bit :](#_bookmark58) 8 bit data to 8 Linear conversion of bit data.
* [RK\_MPI\_IVE\_OrdStatFilter :](#_bookmark59) Image median filtering, minimum filtering, and maximum filtering.
* [RK\_MPI\_IVE\_Map :](#_bookmark61) Image pixel value mapping is performed according to the lookup table.
* [RK\_MPI\_IVE\_EqualizeHist :](#_bookmark62) Image histogram equalization.
* [RK\_MPI\_IVE\_NCC :](#_bookmark63) Normalized cross-correlation coefficient between two grayscale images of the same resolution.
* [RK\_MPI\_IVE\_CCL :](#_bookmark64) Connected component labeling for binary images.
* [RK\_MPI\_IVE\_GMM :](#_bookmark66) Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG .​
* [RK\_MPI\_IVE\_GMM2 :](#_bookmark67) Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG2 .​
* [RK\_MPI\_IVE\_CannyEdge :](#_bookmark68) Extract edge information from grayscale images.
* [RK\_MPI\_IVE\_LBP :](#_bookmark70) Calculate image LBP feature.
* [RK\_MPI\_IVE\_NormGrad :](#_bookmark71) Image normalized gradient calculation, all gradient components are normalized to S8 .
* [RK\_MPI\_IVE\_LKOpticalFlowPyr :](#_bookmark72) LK Optical flow calculation (external pyramid building) .
* [RK\_MPI\_IVE\_LKOpticalFlow :](#_bookmark73) LK Optical flow calculation (building a pyramid internally) .
* [RK\_MPI\_IVE\_STCandiCorner :](#_bookmark74) The first step of image corner detection is to calculate the corresponding intensity of the corner points and filter the corner points.
* [RK\_MPI\_IVE\_STCorner](#_bookmark75) : The second step of image corner detection is to sort the candidate corner points according to the rules.
* [RK\_MPI\_IVE\_MatchBgModel :](#_bookmark76) Based on CODEBOOK The first step of background subtraction is background model training.
* [RK\_MPI\_IVE\_UpdateBgModel :](#_bookmark77) Based on CODEBOOK In the second step of the background subtraction operation, the background model is updated.
* RK\_MPI\_IVE\_SAD : Calculate the two images in 4X4\8X8\16X16 Blocked 16 Bit \8 SAD​ Image, and SAD Threshold the output.
* [RK\_MPI\_IVE\_Warp\_Affine\_Init :](#_bookmark79) Initialize affine transformation auxiliary memory.
* [RK\_MPI\_IVE\_Warp\_Affine :](#_bookmark80) Perform image affine transformation tasks.
* [RK\_MPI\_IVE\_Pyramid\_GetSize :](#_bookmark81) Gets the auxiliary memory size required to generate the image pyramid.
* [RK\_MPI\_IVE\_Pyramid\_Create :](#_bookmark82) Executes the task of creating an image pyramid.
* [RK\_MPI\_IVE\_Query :](#_bookmark83) Query the completion status of created tasks.

#### RK\_MPI\_IVE\_Init

Complete IVE Context creation and necessary initialization

RK\_S32 **RK\_MPI\_IVE\_Init** ();

Return value **:**

0

成功。

非 0

失败，参见错误码。

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* External applications using IVE Before calling a series of operators, call this interface in the program initialization part to complete the IVE

Context creation and necessary initialization work.

* IVE operator interface directly without calling this interface for initialization , the system will automatically complete the initialization. However, if a large number of operators are called, initialization and memory allocation will be performed frequently, resulting in a waste of system resources.
* After finishing using IVE related functions, you must call RK\_MPI\_IVE\_Deinit to perform deinitialization to prevent system resources from being released.

#### RK\_MPI\_IVE\_Deinit

Destroy IVE Context, release related system resources

RK\_S32 **RK\_MPI\_IVE\_Deinit** ();

Return value:

0

成功。

非 0

失败，参见错误码。

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* External applications no longer use IVE After that, when the program exits, call this interface to destroy IVE Context.
* If this interface is not called for initialization, the system will automatically complete the deinitialization work. However, when a large number of operators are called, deinitialization and memory release will be performed frequently, resulting in a waste of system resources.
* Before using this interface, make sure RK\_MPI\_IVE\_Init has been called. Complete IVE Initialization work.

#### RK\_MPI\_IVE\_CvtImageToData

A single PLANAR ImageIVE\_IMAGE\_S​ Type conversion to IVE\_DATA\_S type

RK\_S32 **RK\_MPI\_IVE\_CvtImageToData** [( IVE\_IMAGE\_S](#_bookmark85) \*image,

[IVE\_DATA\_S](#_bookmark88) \*data);

"parameter":

|  |  |  |
| --- | --- | --- |
| image | Input image pointer.  Cannot be empty. | enter |
| data | Output 2D data pointer.  Cannot be empty. | Output |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| image | IVE\_IMAGE\_S | 1 byte | 1x1~2047x2047 |
| data | IVE\_DATA\_S | 1 byte | 1x1~2047x2047 |

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* In this mode, the structure type conversion only supports a single planar input image type. image.

#### RK\_MPI\_IVE\_CvtDataToImage

The two-dimensional data type IVE\_DATA\_S Convert to a single planar ImageIVE\_IMAGE\_S​ type.

RK\_S32 **RK\_MPI\_IVE\_CvtDataToImage** (IVE\_DATA\_S \*data,

IVE\_IMAGE\_S \*image);

"parameter":

|  |  |  |
| --- | --- | --- |
| data | Input 2D data pointer.  Cannot be empty. | enter |
| image | Output image pointer.  Cannot be empty. | Output |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| data | IVE\_DATA\_S | 1 byte | 1x1~2047x2047 |
| image | IVE\_IMAGE\_S | 1 byte | 1x1~2047x2047 |

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* In this mode, the output image type is a single planar image.

#### RK\_MPI\_IVE\_CvtImageToMemInfo

A single PLANAR ImageIVE\_IMAGE\_S​ Type conversion to the one-dimensional data type IVE\_MEM\_INFO\_S .

RK\_S32 **RK\_MPI\_IVE\_CvtImageToMemInfo** (IVE\_IMAGE\_S \*image,

IVE\_MEM\_INFO\_S \*mem);

"parameter":

|  |  |  |
| --- | --- | --- |
| image | Output image pointer.  Cannot be empty. | enter |
| mem | Output one-dimensional data pointer.  Cannot be empty. | Output |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| image | IVE\_IMAGE\_S | 1 byte | 1x1~2047x2047 |
| mem | IVE\_MEM\_INFO\_S | 1 byte | 1x1~2047x2047 |

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* In this mode, the structure type conversion only supports a single planar input image type. image.

### RK\_MPI\_IVE\_DMA

Direct memory access, supporting fast copy, interval copy, and memory filling

RK\_S32 **RK\_MPI\_IVE\_DMA** (IVE\_HANDLE \*pHandle,

[IVE\_DATA\_S](#_bookmark88) \*pstSrc, [IVE\_DST\_DATA\_S](#_bookmark90) \*pstDst, [IVE\_DMA\_CTRL\_S](#_bookmark85) \*pstDmaCtrl, bool bInstant);

"parameter":

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | Input ( also output in set mode ) |
| pstDst | Output data pointer.  Cannot be empty in copy mode | Output |
| pstDmaCtrl | DMA control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Note: Copy Mode refers to IVE\_DMA\_MODE\_DIRECT\_COPY and IVE\_DMA\_MODE\_INTERVAL\_COPY model;

Set Mode refers to IVE\_DMA\_MODE\_SET\_3BYTE and IVE\_DMA\_MODE\_SET\_8BYTE mode.

Return value:

0

成功。

非 0

失败，参见错误码。

"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | [IVE\_DATA\_S](#_bookmark88) | 1 byte | 1x1~2047x2047 |
| pstDst | [IVE\_DATA\_S](#_bookmark88) | 1 byte | 1x1~2047x2047 |

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* IVE\_DMA\_MODE\_DIRECT\_COPY : Fast copy mode allows you to deduct small blocks of memory from large blocks of memory .
* IVE\_DMA\_MODE\_INTERVAL\_COPY : Interval copy mode
* The source data width is required to be u8HorSegSize multiples of;
* Interval copy method: Split the first row of data in each u8VerSegRows row into segments of u8HorSegSize size , and copy the first u8ElemSize size bytes in each segment .
* IVE\_DMA\_MODE\_SET\_3BYTE : 3 Byte stuffing mode
* Use only pstSrc and u64Val Low 3 Bytes are used to fill the source data; when the end of a line is not enough for 3 Bytes , fill with the low byte of u64Val .
* IVE\_DMA\_MODE\_SET\_8BYTE : 8 Byte stuffing mode
* Use only pstSrc and u64Val Fill the source data; when the end of a line is less than 8 Bytes, use

u64Val The low byte is filled.

#### RK\_MPI\_IVE\_Filter

5X5 Template filtering tasks can achieve different filtering by configuring different template coefficients.

RK\_S32 **RK\_MPI\_IVE\_Filter** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_FILTER\_CTRL\_S](#_bookmark95) \*pstFltCtrl, bool bInstant);

"parameter":

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstFltCtrl | Filter Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 ,  YUV420SP , YUV422SP | 1 byte | 5x5~2047x2047 |
| pstDst | U8C1 ,  YUV420SP , YUV422SP | 1 byte | 5x5~2047x2047 |

Quote:

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Filter The calculation formula is as follows :

𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)} ≫ 𝑛𝑜𝑟𝑚

−2≤𝑗≤2 −2≤𝑖≤2

### RK\_MPI\_IVE\_CSC

Color space conversion, can achieve YUV , RGB , HSV Color space conversion.

RK\_S32 **RK\_MPI\_IVE\_CSC** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_CSC\_CTRL\_S](#_bookmark97) \*pstCscCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstCscCtrl | CSC Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 , U8C3\_PACKAGE ,  U8C3\_PLANAR 、  YUV420SP , YUV422SP | 1 byte | 2x2~2046x2046 |
| pstDst | U8C1 , U8C3\_PACKAGE ,  U8C3\_PLANAR 、  YUV420SP , YUV422SP | 1 byte | 2x2~2046x2046 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* When the output image is in multi-plane format, the output data span must be consistent.

#### RK\_MPI\_IVE\_Sobel

3X3 or 5X5 Template SOBEL-LIKE Gradient calculation.

RK\_S32 **RK\_MPI\_IVE\_Sobel** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \* pstDstH, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \* pstDstV , [IVE\_SOBEL\_CTRL\_S](#_bookmark98) \*pstSobelCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDstH | The horizontal image pointer obtained after filtering.  Cannot be empty. | Output |
| pstDstV | The vertical image pointer obtained after filtering.  Cannot be empty. | Output |
| pstCscCtrl | CSC Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2047x2047 |
| pstDstH | S16C1 | 1 byte | 5x5~2047x2047 |
| pstDstV | S16C1 | 1 byte | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Sobel The calculation formula is as follows:

𝐻 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)} ≫ 𝑛𝑜𝑟𝑚

−2≤𝑗≤2 −2≤𝑖≤2

𝑉 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)} ≫ 𝑛𝑜𝑟𝑚

−2≤𝑗≤2 −2≤𝑖≤2

#### RK\_MPI\_IVE\_MagAndAng

Calculate the amplitude and angle

RK\_S32 **RK\_MPI\_IVE\_MagAndAng** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDstMag, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDstAng,

[IVE\_MAG\_AND\_ANG\_CTRL\_S](#_bookmark99) \*pstMagAndAngCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDstMag | Pointer to the output magnitude image.  Cannot be empty. | Output |
| pstDstAng | Output argument image pointer.  Cannot be empty. | Output |
| pstMagAndAngCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2047x2047 |
| pstDstMag | U16C1 | 1 byte | 5x5~2047x2047 |
| pstDstAng | U8C1 | 1 byte | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The output type of magnitude 2D data is IVE\_IMAGE\_TYPE\_U16C1 , and the output type of argument 2D data is IVE\_IMAGE\_TYPE\_U8C1 .
* The calculation formula of the image gradient amplitude is as follows :

𝐻 𝑜𝑢𝑡 = ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)

−2≤𝑗≤2 −2≤𝑖≤2

𝑉 𝑜𝑢𝑡 = ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑗, 𝑖)

−2≤𝑗≤2 −2≤𝑖≤2

𝑀𝑎𝑔 ( 𝑥, 𝑦 ) = 𝑎𝑏𝑠(𝐻 𝑜𝑢𝑡 ( igh, 𝑦 ) ) + 𝑎𝑏𝑠(𝑉 𝑜𝑢𝑡 ( igh, 𝑦 ) )

Where 𝐼 ( 𝑥 , 𝑦 ) is the input source image, 𝑐𝑜𝑒𝑓(𝑚𝑎𝑠𝑘) is the 5x5 template coefficient for calculating the gradient , 𝐻 𝑜𝑢𝑡 is the horizontal image gradient, and 𝑉 𝑜𝑢𝑡 is the vertical image gradient. 𝑀𝑎𝑔 ( 𝑥 , 𝑦 ) is the image gradient amplitude.

* The formula for calculating the argument is as follows :

𝐻 𝑜𝑢𝑡 = ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)

−2≤𝑗≤2 −2≤𝑖≤2

𝑉 𝑜𝑢𝑡 = ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑗, 𝑖)

−2≤𝑗≤2 −2≤𝑖≤2

𝜃 ( 𝑥, 𝑦 ) = 𝑎𝑟𝑐𝑡𝑎𝑛 ( 𝑉 𝑜𝑢 𝑡 )

𝐻 𝑜𝑢𝑡

|  |  |
| --- | --- |
|  | Where I(x,y) is the input source image, coef(mask) is the 5x5 template coefficient for calculating the gradient , H\_out is the horizontal image gradient, and V\_out is the vertical image gradient. θ(x,y) is the image gradient amplitude. Angle, argument angle correspond to 0~7 in the figure according to the calculation result data  The direction value of . |

Figure 5-1​

#### RK\_MPI\_IVE\_Dilate

Image dilation, image morphological operations

RK\_S32 **RK\_MPI\_IVE\_Dilate** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_DILATE\_CTRL\_S](#_bookmark37) \*pstDilateCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstDilateCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2047x2047 |
| pstDst | U8C1 | 1 byte | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image expansion calculation formula:

I out (x, y) = max(I(x − 2, y − 2)&coef(−2, −2), I(x − 1, y − 2)&coef(−1, −2), . . . , I(x + 2, y + 2)&coef(2,2))

#### RK\_MPI\_IVE\_Erode

Image erosion, image morphological operations.

RK\_S32 **RK\_MPI\_IVE\_Erode** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_ERODE\_CTRL\_S](#_bookmark101) \*pstErodeCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstErodeCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2047x2047 |
| pstDst | U8C1 | 1 byte | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image corrosion calculation formula:

I out (x, y) = (I(x − 2, y − 2)&coef(−2, −2)) & (I(x − 1, y − 2)&coef(−1, −2)) & . . . & (I(x + 2, y + 2)&coef(2,2))

#### RK\_MPI\_IVE\_Add

Weighted addition operation of two grayscale images

RK\_S32 **RK\_MPI\_IVE\_Add** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_ADD\_CTRL\_S](#_bookmark103) \*pstAddCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstAddCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image weighted addition calculation formula:

I out (i, j) = ((x ∗ I src1 (i, j) + y ∗ I src2 (i, j)) ≫ 16)&0xff

#### RK\_MPI\_IVE\_And

AND operation between two grayscale images

RK\_S32 **RK\_MPI\_IVE\_And** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 Binary graph | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 Binary graph | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 Binary graph | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image phase and calculation formula:

I out (i, j) = I src1 (i, j) & I src2 (i, j)

#### RK\_MPI\_IVE\_Sub

Subtraction operation between two grayscale images

RK\_S32 **RK\_MPI\_IVE\_Sub** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_SUB\_CTRL\_S](#_bookmark104) \*pstSubCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 , S8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image subtraction calculation formula:

I out (i, j) = abs(I src1 (i, j) − I src2 (i, j))

#### RK\_MPI\_IVE\_Or

OR operation between two grayscale images

RK\_S32 **RK\_MPI\_IVE\_Or** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image phase or calculation formula:

I out (i, j) = I src1 (i, j) | I src2 (i, j)

#### RK\_MPI\_IVE\_Xor

Differentiate or perform operations on two grayscale images

RK\_S32 **RK\_MPI\_IVE\_Xor** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image difference calculation formula:

I out (i, j) = I src1 (i, j)^ I src2 (i, j)

#### RK\_MPI\_IVE\_Integ

Image integral map calculation

RK\_S32 **RK\_MPI\_IVE\_Integ** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_INTEG\_CTRL\_S](#_bookmark105) \*pstIntegCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstIntegCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstDst | U32C1 , U64C1 | 16 bytes | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Sum integral graph ( U32C1 ): mode IVE\_INTEG\_OUT\_CTRL\_SUM , the calculation formula is as follows:

i≤x j≤y

I sum (x, y) = ∑ ∑ I(i, j)

i≥0 j≥0

The integral value at I sum (x , y) is the sum of the grayscale values from (0,0) to (x,y) .

Its format in memory is as follows:

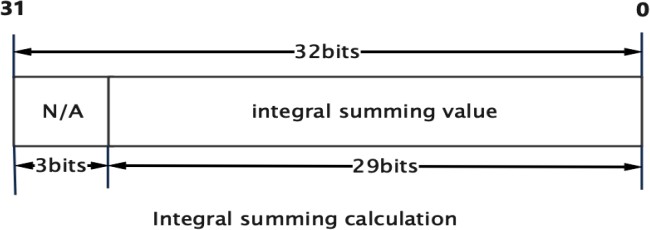


Figure 5-2​

* Square sum integral graph ( U64C1 ): mode IVE\_INTEG\_OUT\_CTRL\_SQSUM , the calculation formula is as follows :

i≤x j≤y

I sq (x, y) = ∑ ∑(I(i, j) ∗ I(i, j))

i≥0 j≥0

The integral value at I sq (x , y) is the sum of the squares of the grayscale values from (0,0) to (x,y) .

Its format in memory is as follows:

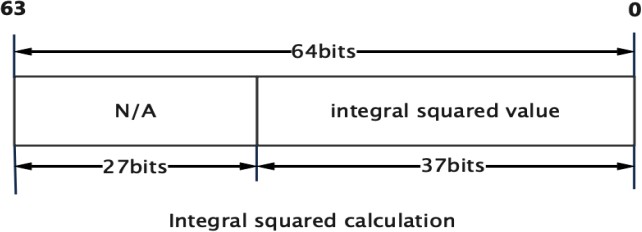


Figure 5-3​

* Sum integral and square sum integral combined ( U64C1 ): mode IVE\_INTEG\_OUT\_CTRL\_COMBINE , the calculation formula is as follows:

I OUT (X, Y) = (I SQ (X, Y) ≪ 28)) | (I SUM (X, Y)&0XFFFFFF)

The memory format of the sum integral graph and square sum integral graph is as follows:

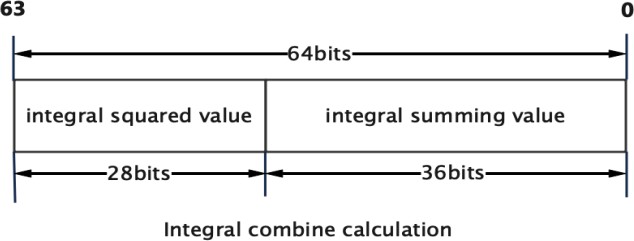


Figure 5- 4

* Auxiliary memory required for connected component calculations.

IVE\_INTEG\_OUT\_CTRL\_SUM mode requires at least 3\*3 of the input image height to be allocated. IVE\_INTEG\_OUT\_CTRL\_SQSUM mode requires at least 4\*4 of the input image height to be allocated. IVE\_INTEG\_OUT\_CTRL\_COMBINE Mode needs to allocate at least input image height \*6 Size of memory

#### RK\_MPI\_IVE\_Hist

Image histogram calculation

RK\_S32 **RK\_MPI\_IVE\_Hist** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstDst,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output histogram pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 16 bytes | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Histogram calculation formula:

hist(x) = ∑∑((I(i, j) == x)? 1: 0) x = 0. . .255

i j

#### RK\_MPI\_IVE\_Thresh

Image 8 Grayscale image binarization operation

RK\_S32 **RK\_MPI\_IVE\_Thresh** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_THRESH\_U8\_CTRL\_S](#_bookmark106) \*pstThrCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstThrCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image threshold binarization supports BINARY , TRUNC , MINVAL , MIN\_MIN\_MAX , ORI\_MID\_MAX , MIN\_MIN\_ORI , MIN\_ORI\_MAX , ORI\_MID\_ORI Mode, the calculation formula is as follows :
* BINARY:

𝐼 𝑜𝑢𝑡

(𝑥, 𝑦) = { 𝑚　𝑛𝑉𝑎𝑙 (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟)

𝑚𝑎igh𝑉𝑎𝑙 (𝐼(igh, 𝑦) > 𝑙𝑜𝑤𝑇ℎ𝑟)

* TRUNC:
* MINVAL:

𝐼 𝑜𝑢𝑡

𝐼

(𝑥, 𝑦) = { 𝐼(𝑥, 𝑦) (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟)

𝑚𝑎igh𝑉𝑎𝑙 (𝐼(igh, 𝑦) > 𝑙𝑜𝑤𝑇ℎ𝑟)

(𝑥, 𝑦) = { 𝑚　𝑛𝑉𝑎𝑙 (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟)

𝑜𝑢𝑡

𝐼(𝑥, 𝑦) (𝐼(𝑥, 𝑦) > 𝑙𝑜𝑤𝑇ℎ𝑟)

|  |  |  |
| --- | --- | --- |
| * MIN\_MID\_MAX: |  | |
|  | 𝑚𝑖𝑛𝑉𝑎𝑙 | (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟) |
|  | 𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { 𝑚𝑖𝑑𝑉𝑎𝑙  𝑚𝑎𝑥𝑉𝑎𝑙 | (𝑙𝑜𝑤𝑇ℎ𝑟 < 𝐼(𝑥, 𝑦) ≤ ℎ𝑖𝑔ℎ𝑇ℎ𝑟)  (𝐼(𝑥, 𝑦) > ℎ𝑖𝑔ℎ𝑇ℎ𝑟) |
| * ORI\_MID\_MAX: |  |  |
|  | 𝐼(𝑥, 𝑦) | (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟) |
|  | 𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { 𝑚𝑖𝑑𝑉𝑎𝑙  𝑚𝑎𝑥𝑉𝑎𝑙 | (𝑙𝑜𝑤𝑇ℎ𝑟 < 𝐼(𝑥, 𝑦) ≤ ℎ𝑖𝑔ℎ𝑇ℎ𝑟)  (𝐼(𝑥, 𝑦) > ℎ𝑖𝑔ℎ𝑇ℎ𝑟) |

|  |  |  |
| --- | --- | --- |
| * MIN\_MIN\_ORI: |  | |
|  | 𝑚𝑖𝑛𝑉𝑎𝑙 | (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟) |
|  | 𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { 𝑚𝑖𝑑𝑉𝑎𝑙  𝐼(𝑥, 𝑦) | (𝑙𝑜𝑤𝑇ℎ𝑟 < 𝐼(𝑥, 𝑦) ≤ ℎ𝑖𝑔ℎ𝑇ℎ𝑟)  (𝐼(𝑥, 𝑦) > ℎ𝑖𝑔ℎ𝑇ℎ𝑟) |
| * MIN\_ORI\_MAX: |  |  |
|  | 𝑚𝑖𝑛𝑉𝑎𝑙 | (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟) |
|  | 𝐼 𝑜𝑢𝑡 ( igh, 𝑦 ) = { 𝐼(𝑥, 𝑦)  𝑚𝑎𝑥𝑉𝑎𝑙 | (𝑙𝑜𝑤𝑇ℎ𝑟 < 𝐼(𝑥, 𝑦) ≤ ℎ𝑖𝑔ℎ𝑇ℎ𝑟)  (𝐼(𝑥, 𝑦) > ℎ𝑖𝑔ℎ𝑇ℎ𝑟) |
| * ORI\_MID\_ORI: |  |  |
|  | 𝐼(𝑥, 𝑦) | (𝐼(𝑥, 𝑦) ≤ 𝑙𝑜𝑤𝑇ℎ𝑟) |
|  | 𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { 𝑚𝑖𝑑𝑉𝑎𝑙  𝐼(𝑥, 𝑦) | (𝑙𝑜𝑤𝑇ℎ𝑟 < 𝐼(𝑥, 𝑦) ≤ ℎ𝑖𝑔ℎ𝑇ℎ𝑟)  (𝐼(𝑥, 𝑦) > ℎ𝑖𝑔ℎ𝑇ℎ𝑟) |

#### RK\_MPI\_IVE\_Thresh\_u16

16 bit data to 8 Bit data threshold binarization operation.

RK\_S32 **RK\_MPI\_IVE\_Thresh\_U16** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_THRESH\_U16\_CTRL\_S](#_bookmark106) \*pstThrCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstThrCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U16C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image threshold binarization supports IVE\_THRESH\_U16\_MODE\_U16\_TO\_U8\_MIN\_MID\_MAX and IVE\_THRESH\_U16\_MODE\_U16\_TO\_U8\_MIN\_ORI\_MAX modes .

#### RK\_MPI\_IVE\_Thresh\_s16

16 with sign bit bit data to 8 Threshold binarization operation of bit data

RK\_S32 **RK\_MPI\_IVE\_CSC** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_THRESH\_U16\_CTRL\_S](#_bookmark106) \*pstThrCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstThrCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | S16C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 , S8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image threshold binarization supports: IVE\_THRESH\_S16\_MODE\_S16\_TO\_S8\_MIN\_MID\_MAX , IVE\_THRESH\_S16\_MODE\_S16\_TO\_S8\_MIN\_ORI\_MAX , IVE\_THRESH\_S16\_MODE\_S16\_TO\_U8\_MIN\_MID\_MAX , IVE\_THRESH\_S16\_MODE\_S16\_TO\_U8\_MIN\_ORI\_MAX .

#### RK\_MPI\_IVE\_16bitto8bit

16 bit data to 8 Linear conversion of bit data

RK\_S32 **RK\_MPI\_IVE\_CSC** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst,

[IVE16BIT\_TO\_8BIT\_CTRL\_S](#_bookmark108) \*pst16BitTo8BitCtrl,

bool bInstant);

"parameter":

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pst16BitTo8BitCt rl | CSC Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U16C1 , S16C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 , S8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Supports 4 modes: IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_S8 IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_ABS IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_BIAS IVE\_16BIT\_TO\_8BIT\_MODE\_U16\_TO\_U8

#### RK\_MPI\_IVE\_8bitto8bit

8 bit data to 8 Linear conversion of bit data

RK\_S32 **RK\_MPI\_IVE\_8BitTo8Bit** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_8BIT\_TO\_8BIT\_CTRL\_S](#_bookmark107) \*pst8BitTo8BitCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pst8BitTo8BitCtr l | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 , S8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 , S8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Supports 4 modes: IVE\_8BIT\_TO\_8BIT\_MODE\_S16\_TO\_S8 IVE\_8BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_ABS IVE\_8BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_BIAS IVE\_8BIT\_TO\_8BIT\_MODE\_U16\_TO\_U8

#### RK\_MPI\_IVE\_OrdStatFilter

Image median filtering, minimum filtering, maximum filtering

RK\_S32 **RK\_MPI\_IVE\_OrdStatFilter** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst,

[IVE\_ORD\_STAT\_FILTER\_CTRL\_S](#_bookmark109) \*pstOrdStatFltCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstOrdStatFilterCtrl | CSC Control parameter pointer .  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 16 bytes | 5x5~2047x2047 |
| pstDst | U8C1 | 16 bytes | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

Calculation formulas for three filtering modes:

* + IVE\_ORD\_STAT\_FILTER\_MODE\_MEDIAN :

𝐼 𝑜𝑢𝑡 (𝑥, 𝑦) = 𝑚𝑒𝑑ade𝑎𝑛 −1≤　≤1 { 𝐼(𝑥 + 𝑖, 𝑦 + 𝑗) }

−1≤𝑗≤1

* + IVE\_ORD\_STAT\_FILTER\_MODE\_MAX :

𝐼 𝑜𝑢𝑡 (𝑥, 𝑦) = 𝑚𝑎igh −1≤　≤1 { 𝐼(igh + 𝑖, 𝑦 + 𝑗) }

−1≤𝑗≤1

* + IVE\_ORD\_STAT\_FILTER\_MODE\_MIN :

𝐼 𝑜𝑢𝑡 (𝑥, 𝑦) = 𝑚　𝑛 −1≤　≤1 { 𝐼(𝑥 + 𝑖, 𝑦 + 𝑗) }

−1≤𝑗≤1

#### RK\_MPI\_IVE\_Map

Image pixel value mapping is performed according to the lookup table.

RK\_S32 **RK\_MPI\_IVE\_Map** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_SRC\_MEM\_INFO\_S](#_bookmark92) \*pstMap [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_MAP\_CTRL\_S](#_bookmark110) \*pstMapCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstMap | Lookup table cache pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstMapCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 1x1~2047x2047 |
| pstMap | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 , U16C1 , S16C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The calculation formula is as follows :

𝐼 𝑜𝑢𝑡 (𝑥, 𝑦) = 𝑚𝑎𝑝[𝐼(𝑥, 𝑦)]

#### RK\_MPI\_IVE\_EqualizeHist

Image histogram equalization.

RK\_S32 **RK\_MPI\_IVE\_EqualizeHist** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_EQHIST\_CTRL\_S](#_bookmark111) \*pstEqualizeHistCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstEqualizeHistCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value:

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Control parameter pstEqualizeHistCtrl u32HistMem​ For auxiliary memory, you need to allocate at least 256 \* sizeof(RK\_U32) size.

#### RK\_MPI\_IVE\_Ncc

Normalized cross-correlation coefficient between two grayscale images of the same resolution.

RK\_S32 **RK\_MPI\_IVE\_NCC** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstDst,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstDst | Output data pointer.  Cannot be empty. | Output |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 1x1~2047x2047 |
| pstDst | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The calculation formula is as follows:

∑ w ∑ h

(I src1 (i, j) ∗ I src2 (i, j))

NCC(I

src1

, I src2 ) =

√∑w

i=1 j=1

∑h (I2 (i, j)) √ ∑w ∑h

(I 2 (i, j))

i=1

j=1

src1

i=1

j=1 src2

### RK\_MPI\_IVE\_CCL

Connected component labeling for binary images.

RK\_S32 **RK\_MPI\_IVE\_CCL** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \* pstSrcDst, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstBlob, [IVE\_CCL\_CTRL\_S](#_bookmark112) \*pstCclCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrcDst | Source image pointer, connected areas are also marked on the source image output .  Cannot be empty. | Input and Output |
| pstBlob | Connected area information pointer.  Cannot be empty. | Output |
| pstCclCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrcDst | U8C1 | 1 byte | 64x64~1984x2047 |
| pstBlob | U8C1 | 16 bytes | 64x64~1984x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The information of the connected area is stored in pstBlob→astRegion Its format in memory is shown in the figure below :

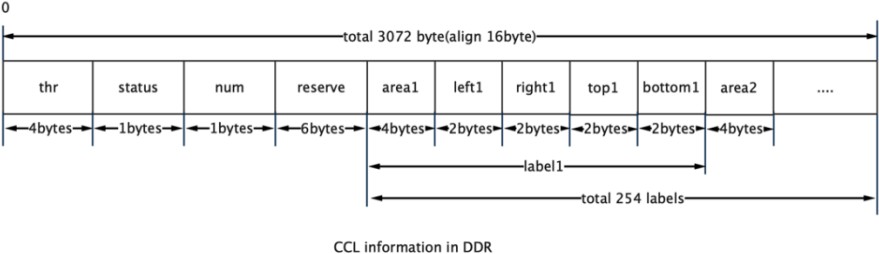


Figure 5- 5

#### RK\_MPI\_IVE\_Gmm

Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG .​

RK\_S32 **RK\_MPI\_IVE\_GMM** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstFg, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstBg, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstMatchModelInfo, [IVE\_MEM\_INFO\_S](#_bookmark91) \*pstModel, [IVE\_GMM\_CTRL\_S](#_bookmark116) \*pstGmmCtrl,

bool bInstant);

"parameter":

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Pointer to the source image.  Cannot be empty. | enter |
| pstFg | Pointer to the foreground image.  Cannot be empty. | Output |
| pstBg | Background image pointer.  Cannot be empty. | Output |
| pstMatchModelInfo | Match information pointer.  Cannot be empty. | Input and Output |
| pstModel | Pointer to Gaussian mixture model.  Cannot be empty. | Input and Output |
| pstGmmCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 , U8C3\_PACKAGE | 16 bytes | 1x1~2047x2047 |
| pstFg | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstBg | U8C1 , U8C3\_PACKAGE | 16 bytes | 1x1~2047x2047 |
| pstMatchModelInfo | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstModel | [IVE\_MEM\_INFO\_S](#_bookmark91) | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstMatchModelInfo .
* pstModel .
* pstGmmCtrl .

#### RK\_MPI\_IVE\_Gmm2

Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG2 .​

RK\_S32 **RK\_MPI\_IVE\_GMM2** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstFactor, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstFg,

[IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstBg, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstMatchModelInfo, [IVE\_MEM\_INFO\_S](#_bookmark91) \*pstModel, [IVE\_GMM2\_CTRL\_S](#_bookmark117) \*pstGmm2Ctrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstFactor | Model update parameter pointer | enter |
| pstFg | Pointer to the foreground image.  Cannot be empty. | Output |
| pstBg | Background image pointer.  Cannot be empty. | Output |
| pstMatchModelInfo | Match information pointer.  Cannot be empty. | Input and Output |
| pstModel | Pointer to Gaussian mixture model .  Cannot be empty. | Input and Output |
| pstGmm2Ctrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

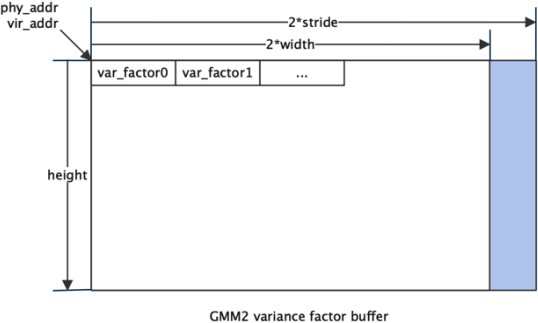
|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 , U8C3\_PACKAGE | 16 bytes | 1x1~2047x2047 |
| pstFactor | U16C1 | 16 bytes | 1x1~2047x2047 |
| pstFg | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstBg | U8C1 , U8C3\_PACKAGE | 16 bytes | 1x1~2047x2047 |
| pstMatchModelInfo | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstModel | [IVE\_MEM\_INFO\_S](#_bookmark91) | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstFactor stores the weight increase ratio after model matching and the threshold for comparing variance changes during model matching. The format in memory is as follows:



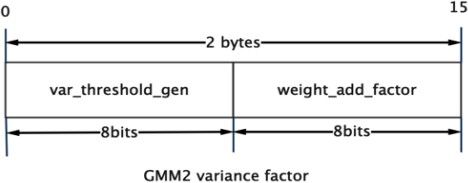


Figure 5-6​

* pstMatchModelInfo stores the model matching flag, model matching index, and number of models. Its in-memory format is as follows:

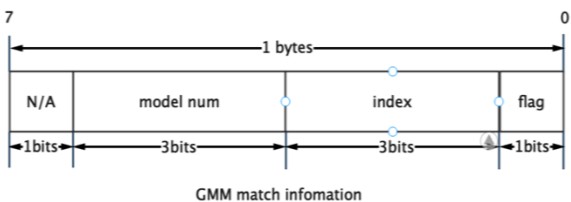
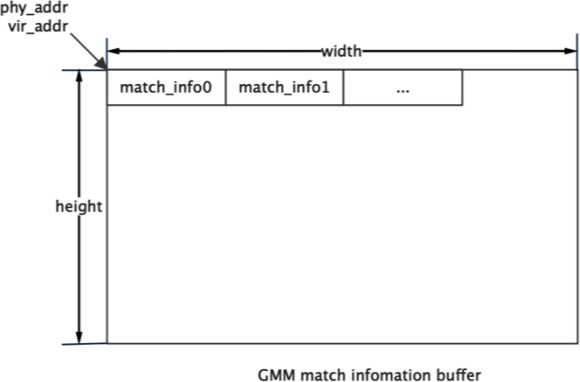


Figure 5-7​

* pstModel Store Gaussian mixture model, when the input image type is grayscale image U8C1 , its format in memory is as follows:

|  |
| --- |
|  |
|  |

Figure 5-8​

When the input image type is RGB888 When an image is created, its format in memory is as follows:

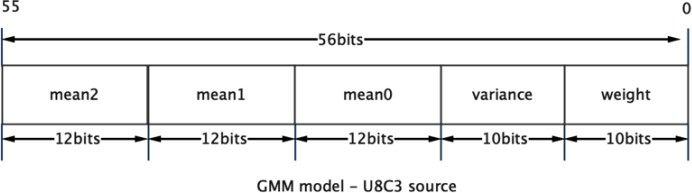
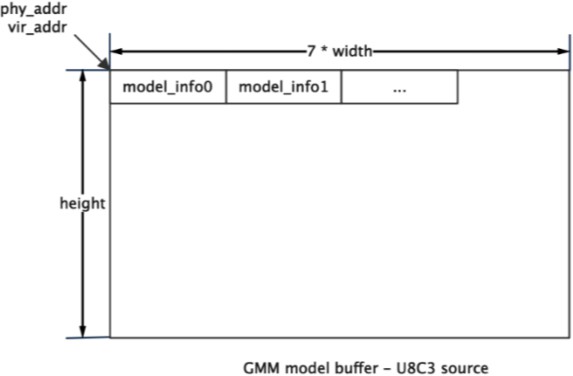


Figure 5-9​

#### RK\_MPI\_IVE\_CannyEdge

Extract edge information from grayscale images.

RK\_S32 **RK\_MPI\_IVE\_CannyEdge** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstEdge, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstStack, [IVE\_CANNY\_EDGE\_CTRL\_S](#_bookmark113) \*pstCannyEdgeCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Pointer to the source image.  Cannot be empty. | enter |
| pstEdge | Edge information image pointer |  |
| pstStack | Edge information pointer. | Output |
| pstCannyEdgeCtr l | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2040x1152 |
| pstEdge | U8C1 | 1 byte | 5x5~2040x1152 |
| pstStack | [IVE\_MEM\_INFO\_S](#_bookmark91) | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstCannyEdgeCtrl stMem​ Allocate at least one-quarter the size of the source image for auxiliary memory.

### RK\_MPI\_IVE\_LBP

Calculate image LBP feature.

RK\_S32 **RK\_MPI\_IVE\_LBP** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst, [IVE\_LBP\_CTRL\_S](#_bookmark115) \*pstLbpCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Pointer to the source image.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstLbpCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 16 bytes | 5x5~2047x2047 |
| pstDst | U8C1 | 16 bytes | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Image local binarization has 2 Modes:

𝑙𝑏𝑝(𝑥, 𝑦) = ∑ 7

𝑖=0

𝑙𝑏𝑝(𝑥, 𝑦) = ∑ 7

( ( 𝐼 walk − 𝐼 𝑐 ) ≥ falconℎ𝑟) ≪ ( 7 − walk ) , where falconℎ𝑟 ∈ [−128,127] (𝑎𝑏𝑠(𝐼− 𝐼 𝑐 ) ≥ falconℎ𝑟 ) ≪ (7 − walk) , where falconℎ𝑟 ∈ [ 0,255]

𝑖=0

#### RK\_MPI\_IVE\_NormGrad

Image normalized gradient calculation, all gradient components are normalized to S8 .

RK\_S32 **RK\_MPI\_IVE\_NormGrad** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDstH, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDstV, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDstHV, IVE\_NORM\_GRAD\_CTRL\_S \*pstNormGradCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Source data pointer.  Cannot be empty. | enter |
| pstDstH | Output horizontal gradient pointer. | Output |
| pstDstV | Output vertical gradient pointer. | Output |
| pstDstHV | Horizontal and vertical gradient pointers. | Output |
| pstNormGradCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 5x5~2047x2047 |
| pstDstH | S8C1 | 1 byte | 5x5~2047x2047 |
| pstDstV | S8C1 | 1 byte | 5x5~2047x2047 |
| pstDstHV | S8C2\_PACKAGE | 1 byte | 5x5~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The calculation formula is as follows:

𝐼 𝑜𝑢𝑡 ( 𝑥, 𝑦 ) = { ∑ ∑ 𝐼 ( 𝑥 + 𝑖, 𝑦 + 𝑗 ) ∗ 𝑐𝑜𝑒𝑓(𝑖, 𝑗)} ≫ 𝑛𝑜𝑟𝑚

−2≤𝑗≤2 −2≤𝑖≤2

#### RK\_MPI\_IVE\_LKOpticalFlowPyr

LK Optical flow calculation (external pyramid building) .

RK\_S32 **RK\_MPI\_IVE\_LKOpticalFlowPyr** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) astSrcPrevPyr[], [IVE\_SRC\_IMAGE\_S](#_bookmark86) astSrcNextPyr[],

[IVE\_SRC\_MEM\_INFO\_S](#_bookmark92) \*pstPrevPts, [IVE\_MEM\_INFO\_S](#_bookmark91) \*pstNextPts, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstStatus, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstErr,

[IVE\_LK\_OPTICAL\_FLOW\_PYR\_CTRL\_S](#_bookmark119) \*pstLkOptiFlowPyrCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| astSrcPrevPyr | The pyramid array of the previous frame image .  Cannot be empty. | enter |
| astSrcNextPyr | The current frame image pyramid array . | enter |
| pstPrevPts | The pointer of the optical flow tracking point in the previous frame . | enter |
| pstNextPts | The optical flow tracking point pointer of the current frame . | enter |
| pstStatus | Tracking status information, 1 Indicates success, 0 Indicates failure. | enter |
| pstErr | Tracking point similarity error estimation .  Cannot be empty. | Output |
| pstLkOptiFlowPyrCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| astSrcPrevPyr[0] , astSrcNextPyr[0] | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstPrevPts[0] ,  pstNextPts[0] | - | 16 bytes | 1x1~2047x2047 |
| pstStatus | - | 16 bytes | - |
| pstErr | - | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstLkOptiFlowPyrCtrl u8MaxLevel​ The value range is [0,3] , and the corresponding number of pyramid levels is [1,4] .
* A pyramid is built externally, requiring the height and width of each layer of image to be half of the height and width of the previous layer of image.

#### RK\_MPI\_IVE\_LKOpticalFlow

LK Optical flow calculation (building a pyramid internally) .

RK\_S32 **RK\_MPI\_IVE\_LKOpticalFlow** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrcPre, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrcCur, [IVE\_SRC\_MEM\_INFO\_S](#_bookmark92) \*pstPoint, [IVE\_SRC\_MEM\_INFO\_S](#_bookmark92) \*pstMv,

[IVE\_LK\_OPTICAL\_FLOW\_CTRL\_S](#_bookmark118) \*pstLkOptiFlowCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrcPre | Pointer to the previous frame image.  Cannot be empty. | enter |
| pstSrcCur | Current frame image pointer.  Cannot be empty. | enter |
| pstPoint | The initial feature point coordinates of the current pyramid layer.  Cannot be empty. | |
| pstMv | Tracking point motion vector pointer.  Cannot be empty. | Output |
| pstLkOptiFlowCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrcPre  pstSrcCur | U8C1 | 16 bytes | 1x1~2047x2047 |
| pstPoint | - | 16 bytes | - |
| pstMv | - | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Internal default build 4 Layer pyramid, the height and width of each layer of image is half of the height and width of the previous layer of image.
* pstMV cache format is IVE\_MV\_S9Q7\_S , which stores tracking status information and tracking point motion vectors. Its format in memory is shown in the figure below:

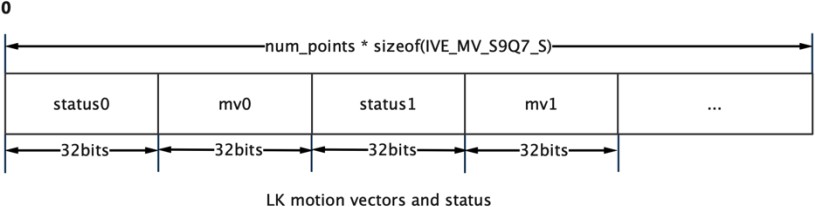


Figure 5-10​

#### RK\_MPI\_IVE\_STCandiCorner

The first step of image corner detection is to calculate the corresponding intensity of the corner points and filter the corner points.

RK\_S32 **RK\_MPI\_IVE\_STCandiCorner** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc,

[IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstCandiCorner, [IVE\_ST\_CANDI\_CORNER\_CTRL\_S](#_bookmark120) \*pstStCandiCornerCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Pointer to the source image.  Cannot be empty. | enter |
| pstCandiCorner | Candidate corner point pointer.  Cannot be empty. | Output |
| pstStCandiCornerCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 16 bytes | 64x64~1280x720 |
| pstCandiCorner | - | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Reference OpenCV Shi -Tomas Corner detection.
* The candidate corner point cache needs to allocate at least u16Width \* u16Height \* sizeof(RK\_U16) + sizeof(IVE\_ST\_CANDI\_STACK\_SIZE\_S) .
* The candidate corner information includes corner response strength, x Coordinates and y The format of coordinates in memory is as follows ;

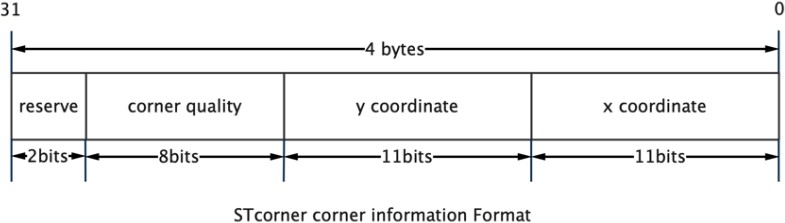


Figure 5-11​

#### RK\_MPI\_IVE\_STCorner

The second step of image corner detection is to sort the candidate corner points according to the rules.

RK\_S32 **RK\_MPI\_IVE\_STCorner** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstCandiCorner, [IVE\_DST\_MEM\_INFO\_S](#_bookmark93) \*pstCorner, [IVE\_ST\_CORNER\_CTRL\_S](#_bookmark121) \*pstStCornerCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc | Pointer to the source image.  Cannot be empty. | enter |
| pstCandiCorner | Candidate corner pointer | enter |
| pstCorner | Output the filtered corner point pointers .  Cannot be empty. | Output |
| pstStCornerCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 16 bytes | 64x64~1280x720 |
| pstCandiCorner | - | 16 bytes | - |
| pstCorner | - | 16 bytes | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstStCornerCtrl stMem​ For auxiliary memory, at least stSrcImg.u32Height needs to be allocated \* stSrcImg.au32Stride[0] + sizeof(IVE\_ST\_CORNER\_MEM\_S) \* 2; .
* The final output corner information is consistent with the format of the candidate corner points in memory. It also includes the corner response strength, x The format of the coordinates and y coordinates in memory is as follows:

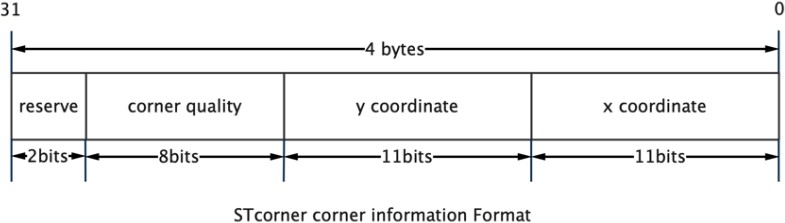


Figure 5-12​

#### RK\_MPI\_IVE\_MatchBgModel

Based on CODEBOOK The first step of background subtraction is background model training.

RK\_S32 **RK\_MPI\_IVE\_MatchBgModel** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstCurImg, [IVE\_DATA\_S](#_bookmark88) \*pstBgModel,

[IVE\_MATCH\_BG\_MODEL\_CTRL\_S](#_bookmark122) \*pstMatchBgModelCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstCurImg | Pointer to the source image.  Cannot be empty. | enter |
| pstBgModel | Background model pointer.  Cannot be empty. | Output |
| pstMatchBgModelCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstCurImg | U8C1 | 1 byte | 1x1~2047x2047 |
| pstBgModel | - | 1 byte | - |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstBgModel Storage book The model format in memory is as follows:

|  |
| --- |
|  |
|  |

Figure 5-13​

#### RK\_MPI\_IVE\_UpdateBgModel

Based on CODEBOOK In the second step of the background subtraction operation, the background model is updated.

RK\_S32 **RK\_MPI\_IVE\_UpdateBgModel** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \* pstCurImg, [IVE\_DATA\_S](#_bookmark88) \*pstBgModel,

[\_IVE\_IMAGE\_S](#_bookmark85) \*pstFgFlag, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstBgImg,

[IVE\_UPDATE\_BG\_MODEL\_CTRL\_S](#_bookmark123) \*pstUpdateBgModelCtrl, bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstCurImg | Pointer to the source image.  Cannot be empty. | enter |
| pstBgModel | Background model pointer.  Cannot be empty. | Input and Output |
| pstFgFlag | Output foreground image pointer | Output |
| pstBgImg | Output background image pointer .  Cannot be empty. | Output |
| pstUpdateBgModelCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time . | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

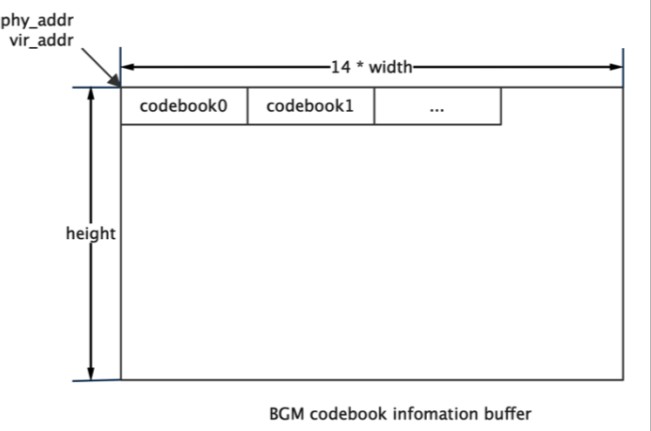
"Require":

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstCurImg | U8C1 | 1 byte | 1x1~2047x2047 |
| pstBgModel | - | 1 byte | - |
| pstFgFlag | U8C1 | 1 byte | 1x1~2047x2047 |
| pstBgImg | U8C1 | 1 byte | 1x1~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* pstBgModel Storage book Model, each model memory size is 14 bytes, whose format in memory is as follows:

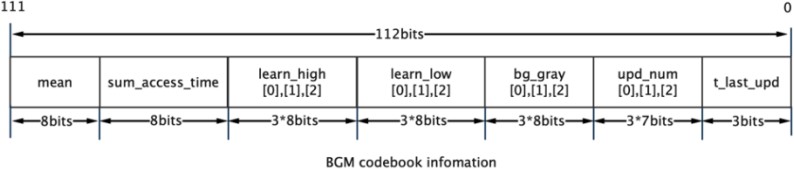


Figure 5- 14

### RK\_MPI\_IVE\_SAD

Calculate the two images according to 4X4\8X8\16X16 Blocked 16 Bit \8 SAD​ Image, and SAD Threshold the output.

RK\_S32 **RK\_MPI\_IVE\_SAD** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \* pstSrc1, [IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc2, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstSad, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstThr, [IVE\_SAD\_CTRL\_S](#_bookmark124) \*pstSadCtrl,

bool bInstant);

"parameter":

|  |  |  |
| --- | --- | --- |
| pHandle | handle pointer.  Cannot be empty. | Output |
| pstSrc1 | Source Image 1 pointer.  Cannot be empty. | enter |
| pstSrc2 | Source Image 2 pointer.  Cannot be empty. | enter |
| pstSad | Output SAD Image pointer.  Cannot be empty. | Output |
| pstThr | Output SAD Thresholded image pointer .  Cannot be empty. | Output |
| pstSadCtrl | Control parameter pointer.  Cannot be empty. | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc1 | U8C1 | 1 byte | 64x64~2047x2047 |
| pstSrc2 | U8C1 | 1 byte | 64x64~2047x2047 |
| pstSad | U8C1 , U16C1 | 1 byte | 64x64~2047x2047 |
| pstThr | U8C1 | 1 byte | 64x64~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The calculation formula is as follows:

Diff(i, j) = |I 1 (i, j) − I 2 (i, j)|

SAD out (x, y) = ∑ Diff(i, j), (x ≥ 0, y ≥ 0, stride = n)

n∗x≤i<n∗(x+1) n∗y≤j<n∗(y+1)

THR

out

(x, y) = { minVal (SAD out (x, y) ≤ Thr) maxVal (SAD out (x, y) > Thr)

in,

n= 4 Corresponding to RVE\_SAD\_MODE\_MB\_4X4 n= 8 Corresponding to RVE\_SAD\_MODE\_MB\_8X8

n=16 Corresponding to RVE\_SAD\_MODE\_MB\_16X16

#### RK\_MPI\_IVE\_Warp\_Affine\_Init

Initialize affine transformation auxiliary memory.

RK\_S32 **RK\_MPI\_IVE\_Warp\_Affine\_Init** [( IVE\_MEM\_INFO\_S](#_bookmark91) \*pstMem,

RK\_U32 u32Width,

RK\_U32 u32Height);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pstMem | Auxiliary memory pointer  Cannot be empty. | Output |
| u32Width | Enter the image width.  Cannot be empty. | enter |
| u32Height | Enter the image height.  Cannot be empty. | Output |

Return value **:**

0

成功。

非 0

失败，参见错误码。

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* 5 times the size of the input image , call this interface to complete the necessary initialization work. Then call RK\_MPI\_IVE\_Warp\_Affine to complete the image affine transformation.

#### RK\_MPI\_IVE\_Warp\_Affine

Perform image affine transformation tasks .

RK\_S32 **RK\_MPI\_IVE\_Warp\_Affine** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc, [IVE\_DST\_IMAGE\_S](#_bookmark87) \*pstDst,

[IVE\_WARP\_AFFINE\_CTRL\_S](#_bookmark125) \*pstWarpAffineCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | Task ID .  Cannot be empty. | Output |
| pstSrc | Input image pointer.  Cannot be empty. | enter |
| pstDst | Output image pointer.  Cannot be empty. | Output |
| pstWarpAffineCtrl | Control parameters | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 16x16~256x256 |
| pstDst | U8C1 | 1 byte | 16x16~256x256 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Before calling this interface, you need to execute RK\_MPI\_IVE\_Warp\_Affine\_Init to complete the necessary initialization work.
* pstWarpAffineCtrl. stAffineMat 3x2​ The affine transformation matrix is as shown below:

[ 𝑎 00 𝑎 01 𝑏 00 ]

𝑎 10 𝑎 11 𝑏 01

in,

𝑎 00 𝑎 01

] is the linear transformation correlation coefficient, and 𝑏 00 ] is the translation correlation coefficient.

[ 𝑎 10

𝑎 11

[

𝑏 01

#### RK\_MPI\_IVE\_Pyramid\_GetSize

Gets the auxiliary memory size required to generate the image pyramid.

RK\_S32 **RK\_MPI\_IVE\_Pyramid\_GetSize** (RK\_U32 u32Width, RK\_U32 u32Height,

[IVE\_PYRAMID\_CTRL\_S](#_bookmark126) \*pstPyramidCtrl);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| u32Width | Enter the image width.  Cannot be empty. | Output |
| u32Height | Enter the image height.  Cannot be empty. | enter |
| pstPyramidCtrl | Control pointer.  Cannot be empty. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* Before applying for auxiliary memory for the image pyramid creation task, call this interface to obtain the required auxiliary memory size.

#### RK\_MPI\_IVE\_Pyramid\_Create

Query the completion status of created tasks.

RK\_S32 **RK\_MPI\_IVE\_Pyramid\_Create** (IVE\_HANDLE \*pHandle,

[IVE\_SRC\_IMAGE\_S](#_bookmark86) \*pstSrc,

[IVE\_DST\_IMAGE\_S](#_bookmark87) pstPyramid[], [IVE\_PYRAMID\_CTRL\_S](#_bookmark126) \*pstPyramidCtrl,

bool bInstant);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | Task ID .  Cannot be empty. | Output |
| pstSrc | Input image pointer.  Cannot be empty. | enter |
| pstPyramid[], | Pyramid image array.  Cannot be empty. | enter |
| pstPyramidCtrl | Control parameters | enter |
| bInstant | Return the result flag in time. | enter |

Return value **:**

0

成功。

非 0

失败，参见错误码。

"Require" **:**

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | Data Type | Address alignment | Resolution |
| pstSrc | U8C1 | 1 byte | 16x16~2047x2047 |
| pstPyramid[] | U8C1 | 1 byte | 16x16~2047x2047 |

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* The default scaling factor of the generated pyramid image is 0.5 , and the relationship between the width and height of the upper and lower pyramid images is:

𝑤𝑖𝑑𝑡ℎ 𝑛+1

ℎ𝑒𝑖𝑔ℎ𝑡 𝑛+1

= (𝑤𝑖𝑑𝑡ℎ 𝑛 + 1)

2

= (ℎ𝑒𝑖𝑔ℎ𝑡 𝑛 + 1)

2

* The default span of pyramid images is 16 Byte alignment.

#### RK\_MPI\_IVE\_Query

Query the completion status of created tasks.

RK\_S32 **RK\_MPI\_IVE\_Query** (IVE\_HANDLE \*pHandle,

bool \*pbFinish,

bool bBlock);

Parameters **:**

|  |  |  |
| --- | --- | --- |
| pHandle | Task ID .  Cannot be empty. | Output |
| pbFinish | Task completion status pointer.  Cannot be empty. | enter |
| bBlock | Whether to block the query flag.  Cannot be empty. | Output |

Return value **:**

0

成功。

非 0

失败，参见错误码。

Quote **:**

|  |  |
| --- | --- |
| header files | rk\_comm\_ive.h , rk\_ive.h , rk\_mpi\_ive.h |
| Library Files | librve.a , librve.so |

"Notice" **:**

* When users use IVE Before the mission is completed, to ensure RKIVE The task has been completed. Users can call this interface to query in blocking mode .

## Data types and structures

**IVE\_IMAGE\_S** **IVE\_SRC\_IMAGE\_S** **IVE\_DST\_IMAGE\_S**

Define 2D image information

"definition"

typedef struct rkIVE\_IMAGE\_S { RK\_U64 au64PhyAddr[3]; RK\_U64 au64VirAddr[3]; RK\_U32 au32Stride[3]; RK\_U32 u32Width;

RK\_U32 u32Height; IVE\_IMAGE\_TYPE\_E enType; RK\_U32 u32Reserved;

**} IVE\_IMAGE\_S;**

typedef IVE\_IMAGE\_S IVE\_SRC\_IMAGE\_S; typedef IVE\_IMAGE\_S IVE\_DST\_IMAGE\_S;

"illustrate"

|  |  |
| --- | --- |
| au64PhyAddr | Image cache physical address array, storing multiple planar Cache first address |
| au64VirAddr | Image cache virtual address array, storing multiple planar Cache first address |
| au32Stride | Image cache line span array, storing multiple planar Row Span |
| u32Width | Image width |
| u32Height | Image Height |
| enType | Image Type |
| u32Reserved | Reserved bits |

**IVE\_DATA\_S** **IVE\_SRC\_DATA\_S** **IVE\_DST\_DATA\_S**

Define two-dimensional data information

"definition"

typedef struct rkIVE\_DATA\_S { RK\_U64 u64PhyAddr; RK\_U64 u64VirAddr;

RK\_U32 u32Stride; RK\_U32 u32Width; RK\_U32 u32Height;

RK\_U32 u32Reserved;

**} IVE\_DATA\_S;**

"illustrate"

typedef IVE\_DATA\_S IVE\_SRC\_DATA\_S; typedef IVE\_DATA\_S IVE\_DST\_DATA\_S;

|  |  |
| --- | --- |
| u64PhyAddr | 2D data cache physical address |
| au64VirAddr | 2D data cache virtual address |
| au32Stride | 2D data cache line span |
| u32Width | 2D data width |
| u32Height | 2D data height |
| u32Reserved | Reserved bits |

**IVE\_MEM\_INFO\_S** **IVE\_SRC\_MEM\_INFO\_S** **IVE\_DST\_MEM\_INFO\_S**

Define one-dimensional data cache information

"definition"

typedef struct rkIVE\_MEM\_INFO\_S { RK\_U64 u64PhyAddr;

RK\_U64 u64VirAddr;

RK\_U32 u32Size;

RK\_U32 u32Reserved;

**} IVE\_MEM\_INFO\_S;**

typedef IVE\_MEM\_INFO\_S IVE\_SRC\_MEM\_INFO\_S; typedef IVE\_MEM\_INFO\_S IVE\_DST\_MEM\_INFO\_S;

|  |  |
| --- | --- |
| u64PhyAddr | One-dimensional data cache physical address |
| au64VirAddr | One-dimensional data cache virtual address |
| u32Size | The memory space occupied by the one-dimensional data cache |
| u32Reserved | Reserved bits |

### IVE\_DMA\_CTRL\_S

Defining DMA Control Information

"definition"

typedef struct rkIVE\_DMA\_CTRL\_S { IVE\_DMA\_MODE\_E enMode; RK\_U64 u64Val;

RK\_U8 u8HorSegSize;

RK\_U8 u8ElemSize;

RK\_U8 u8VerSegRows;

**} IVE\_DMA\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | IVE\_DMA\_MODE\_DIRECT\_COPY: Direct copy mode IVE\_DMA\_MODE\_INTERVAL\_COPY : Interval copy mode IVE\_DMA\_MODE\_SET\_3BYTE: 3- byte fill mode  IVE\_DMA\_MODE\_SET\_8BYTE: 8 Byte stuffing mode |
| u64Val | Only used in fill mode, 3 Byte filling mode uses the lower 3 bytes save. |
| u8HorSegSize | Only used in interval copy mode, the size of the segment that divides the source image into one row horizontally. Value range:  {2, 3, 4, 8, 16} . |
| u8ElemSize | u8ElemSizebyte of each segment is split A valid copy field .  Value range: [1, u8HorSegSize-1] . |
| u8VerSegRows | Only used in interval copy mode, split the first row of data in each u8VerSegRows row into  u8HorSegSize size segment, copy the first u8ElemSize in each segment Size in bytes  Value range: [1, min{65535/srcStride, srcHeight}] . |

### IVE\_FILTER\_CTRL\_S

Define filtering control information

"definition"

typedef struct rkIVE\_FILTER\_CTRL\_S { RK\_U8 u8CoefSel;

RK\_U8 u8Norm;

RK\_U8 u8OutMode;

RK\_S8 as8Mask[25];

**} IVE\_FILTER\_CTRL\_S;**

|  |  |
| --- | --- |
| u8CoefSel | Template coefficient:  0 : 3x3  1 : 5x5 |
| u8Norm | Normalization parameter.  Value range: [0, 13] . |
| u8OutMode | Output data format:  0 : RK\_U8  1 : RK\_S8  2 : RK\_U16  3 : RK\_S16 |
| as8Mask | 5x5 Template coefficient, peripheral coefficient is set to 0 3x3 can be achieved Template filtering. |

### IVE\_CSC\_CTRL\_S

Defining CSC Control Information

"definition"

typedef struct rkIVE\_CSC\_CTRL\_S { IVE\_CSC\_MODE\_E enMode; RK\_U8 u8InDataFmt;

RK\_U8 u8OutDataFmt; RK\_U8 u8YUV2RGBRange; RK\_U8 u8RGB2YUVRange;

**} IVE\_CSC\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | IVE\_CSC\_MODE\_LIMIT\_BT601\_YUV2RGB IVE\_CSC\_MODE\_LIMIT\_BT709\_YUV2RGB IVE\_CSC\_MODE\_FULL\_BT601\_YUV2RGB IVE\_CSC\_MODE\_FULL\_BT709\_YUV2RGB  IVE\_CSC\_MODE\_LIMIT\_BT601\_YUV2HSV IVE\_CSC\_MODE\_LIMIT\_BT709\_YUV2HSV IVE\_CSC\_MODE\_FULL\_BT601\_YUV2HSV IVE\_CSC\_MODE\_FULL\_BT709\_YUV2HSV  IVE\_CSC\_MODE\_LIMIT\_BT601\_RGB2YUV IVE\_CSC\_MODE\_LIMIT\_BT709\_RGB2YUV IVE\_CSC\_MODE\_FULL\_BT601\_RGB2YUV IVE\_CSC\_MODE\_FULL\_BT709\_RGB2YUV  IVE\_CSC\_MODE\_LIMIT\_BT601\_RGB2HSV IVE\_CSC\_MODE\_LIMIT\_BT709\_RGB2HSV IVE\_CSC\_MODE\_FULL\_BT601\_RGB2HSV  IVE\_CSC\_MODE\_FULL\_BT709\_RGB2HSV |
| u8InDataFmt | Input data format. |
| u8OutDataFmt | Output data format. |
| u8YUV2RGBRa  nge | YUV Convert to RGB Mode data range:  0: [16~235]  1: [0~255] |
| u8RGB2YUVRa  nge | RGB Convert to YUV Mode data range:  0: [16~235]  1: [0~255] |

### IVE\_SOBEL\_CTRL\_S

Defining SOBEL Control Information

"definition"

"illustrate"

typedef struct rkIVE\_SOBEL\_CTRL\_S { RK\_U8 u8CoefSel;

RK\_U8 u8OutCtrl;

RK\_U8 u8Norm;

RK\_U8 u8OutMode;

RK\_S8 as8Mask[25];

**} IVE\_SOBEL\_CTRL\_S;**

|  |  |
| --- | --- |
| u8CoefSel | Template coefficient:  0 : 3x3  1 : 5x5 |
| u8OutCtrl | Output Mode:  0 : Horizontal, vertical  1 : Horizontal  2 : Vertical direction |
| u8Norm | Normalization parameter.  Value range: [0, 13] . |
| u8OutMode | Output data format:  0 : RK\_U8  1 : RK\_S8  2 : RK\_U16  3 : RK\_S16 |
| as8Mask | 5x5 Template coefficient, peripheral coefficient is set to 0 3x3 can be achieved Template filtering. |

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

Defines control information for amplitude and angle calculations

"definition"

"illustrate"

typedef struct rkIVE\_MAG\_AND\_ANG\_CTRL\_S { RK\_S8 as8Mask[25];

[IVE\_MEM\_INFO\_S](#_bookmark91) **stMem;**

**} IVE\_MAG\_AND\_ANG\_CTRL\_S;**

|  |  |
| --- | --- |
| as8Mask | 5x5 Template coefficient |
| stMem | Auxiliary memory required for connected area calculations requires at least 4 times the input image buffer size |

### IVE\_DILATE\_CTRL\_S

Define image expansion control information

"definition"

"illustrate"

typedef struct rkIVE\_DILATE\_CTRL\_S { RK\_U8 au8Mask[25];

**} IVE\_DILATE\_CTRL\_S;**

|  |  |
| --- | --- |
| au8Mask | 5x5 Template coefficient |

### IVE\_ERODE\_CTRL\_S

Define image corrosion control information

"definition"

"illustrate"

typedef struct rkIVE\_ERODE\_CTRL\_S { RK\_U8 au8Mask[25];

**} IVE\_ERODE\_CTRL\_S;**

|  |  |
| --- | --- |
| au8Mask | 5x5 Template coefficient |

### IVE\_ADD\_CTRL\_S

Define image weighting and control information

"definition"

"illustrate"

typedef struct rkIVE\_ADD\_CTRL\_S { RK\_U16 u0q16X;

RK\_U16 u0q16Y;

**} IVE\_ADD\_CTRL\_S;**

|  |  |
| --- | --- |
| u0q16 | "x" in the weighted addition "xA+yB" .  Value range: [1, 65535] . |
| u0q16 | Weighted addition "xA+yB" with weight "y" .  Value range: {65536 - u0q16X} . |

### IVE\_SUB\_CTRL\_S

Define image subtraction control information

"definition"

"illustrate"

typedef struct rkIVE\_SUB\_CTRL\_S { IVE\_SUB\_MODE\_E enMode;

**} IVE\_SUB\_CTRL\_S;**

enMode

IVE\_SUB\_MODE\_ABS: 取差的绝对值。

IVE\_SUB\_MODE\_SHIFT: 将结果右移一位输出，保留符号位。

### IVE\_INTEG\_CTRL\_S

Define integral graph control information

"definition"

typedef struct rkIVE\_INTEG\_CTRL\_S { IVE\_INTEG\_OUT\_CTRL\_E enOutCtrl; [IVE\_MEM\_INFO\_S](#_bookmark91) stMem;

**} IVE\_INTEG\_CTRL\_S**

"illustrate"

|  |  |
| --- | --- |
| enOutCtrl | IVE\_INTEG\_OUT\_CTRL\_SUM: Only the integral graph is output.  IVE\_INTEG\_OUT\_CTRL\_SQSUM: Only square and integral plots are output.  IVE\_INTEG\_OUT\_CTRL\_COMBINE : Combined output of sum, square and integral graphs |
| stMem | Auxiliary memory required for connected component calculations.  IVE\_INTEG\_OUT\_CTRL\_SUM mode requires at least input image height \* 3 memory allocation  IVE\_INTEG\_OUT\_CTRL\_SQSUM mode requires at least input image height \* 4 size memory to be allocated  IVE\_INTEG\_OUT\_CTRL\_COMBINE mode requires at least input image height \* 6 memory allocation |

### IVE\_THRESH\_CTRL\_S

Define image threshold binarization control information

"definition"

typedef struct rkIVE\_THRESH\_U8\_CTRL\_S { IVE\_THRESH\_MODE\_E enMode;

RK\_U8 u8LowThr;/ RK\_U8 u8HighThr; RK\_U8 u8MinVal; RK\_U8 u8MidVal; RK\_U8 u8MaxVal;

**} IVE\_THRESH\_U8\_CTRL\_S;**

typedef struct rkIVE\_THRESH\_U16\_CTRL\_S { IVE\_THRESH\_U16\_MODE\_E enMode; RK\_U16 u16LowThr;

RK\_U16 u16HighThr;

RK\_U8 u8MinVal;

RK\_U8 u8MidVal;

RK\_U8 u8MaxVal;

**} IVE\_THRESH\_U16\_CTRL\_S;**

typedef struct rkIVE\_THRESH\_S16\_CTRL\_S { IVE\_THRESH\_S16\_MODE\_E enMode; RK\_S16 S16LowThr;

RK\_S16 S16HighThr;

RK\_S8 S8MinVal;

RK\_S8 S8MidVal;

RK\_S8 S8MaxVal;

**} IVE\_THRESH\_S16\_CTRL\_S;**

"illustrate"

|  |  |
| --- | --- |
| enMode | Thresholding modes : IVE\_THRESH\_MODE\_BINARY IVE\_THRESH\_MODE\_TRUNC IVE\_THRESH\_MODE\_TO\_MINVAL IVE\_THRESH\_MODE\_MIN\_MID\_MAX IVE\_THRESH\_MODE\_ORI\_MID\_MAX IVE\_THRESH\_MODE\_MIN\_MID\_ORI IVE\_THRESH\_MODE\_MIN\_ORI\_MAX  IVE\_THRESH\_MODE\_ORI\_MID\_ORI |
| u8LowThr | Low threshold.  Value range: [0,255] . |
| u8HighThr | High threshold.  0≤u8LowThresh≤u8HighThresh≤255 . |
| u8MinVal | Minimum value.  Value range: [0,255] . |
| u8MidVal | Median value.  Value range: [0,255] . |
| u8MaxVal | Maximum value.  Value range: [0,255] . |

**IVE\_8BIT\_TO\_8BIT\_CTRL\_S** **IVE\_16BIT\_TO\_8BIT\_CTRL\_S**

Definition 8 bit, 16 bit data to 8 bit data linear conversion control information

"definition"

typedef struct rkIVE\_8BIT\_TO\_8BIT\_CTRL\_S { IVE\_8BIT\_TO\_8BIT\_MODE\_E enMode; RK\_U8 u8Denominator;

RK\_U8 u8Numerator;

RK\_S8 s8Bias;

**} IVE\_8BIT\_TO\_8BIT\_CTRL\_S;**

typedef struct rkIVE\_16BIT\_TO\_8BIT\_CTRL\_S { IVE\_16BIT\_TO\_8BIT\_MODE\_E enMode; RK\_U16 u16Denominator;

RK\_U8 u8Numerator;

RK\_S8 s8Bias;

**} IVE\_16BIT\_TO\_8BIT\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | Conversion Mode  U8->U8: IVE\_8BIT\_TO\_8BIT\_MODE\_S8\_TO\_S8 IVE\_8BIT\_TO\_8BIT\_MODE\_S8\_TO\_U8\_ABS  IVE\_8BIT\_TO\_8BIT\_MODE\_S8\_TO\_U8\_BIAS IVE\_8BIT\_TO\_8BIT\_MODE\_U8\_TO\_U8  U16->U8: IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_S8 IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_ABS IVE\_16BIT\_TO\_8BIT\_MODE\_S16\_TO\_U8\_BIAS  IVE\_16BIT\_TO\_8BIT\_MODE\_U16\_TO\_U8 |
| u8Denominator  u16Denominator | The denominator in a linear transformation. |
| u8Numerator | Numerator in a linear transformation.  Value range: [0,255] . |
| s8Bias | Translation term in a linear transformation.  Value range: [-128,127] . |

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

Define order statistics filtering mode

"definition"

typedef struct rkIVE\_ORD\_STAT\_FILTER\_CTRL\_S { IVE\_ORD\_STAT\_FILTER\_MODE\_E enMode;

**} IVE\_ORD\_STAT\_FILTER\_CTRL\_S;**

enMode

IVE\_ORD\_STAT\_FILTER\_MODE\_MEDIAN: 中值滤波

IVE\_ORD\_STAT\_FILTER\_MODE\_MAX: 最大值滤波

IVE\_ORD\_STAT\_FILTER\_MODE\_MIN: 最小值滤波

### IVE\_MAP\_CTRL\_S

Define image filtering mode

"definition"

"illustrate"

typedef struct rkIVE\_MAP\_CTRL\_S { IVE\_MAP\_MODE\_E enMode;

**} IVE\_MAP\_CTRL\_S;**

enMode

IVE\_MAP\_MODE\_U8: U8C1->U8C1 Map 模式

IVE\_MAP\_MODE\_S16: U8C1->U16C1 Map 模式

IVE\_MAP\_MODE\_U16: U8C1->S16C1 Map 模式

### IVE\_EQHIST\_CTRL\_S

Define image histogram equalization control parameters

"definition"

typedef struct rkIVE\_EQUALIZE\_RKST\_CTRL\_S { IVE\_EQUALIZE\_MODE\_E enMode;

RK\_U32 u32HistArray[256];

[IVE\_MEM\_INFO\_S](#_bookmark91) **u32HistMem;**

**} IVE\_EQHIST\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | IVE\_EQUALIZE\_MODE\_EQHIST\_WITH\_EXT\_HIST: External input histogram statistics  IVE\_EQUALIZE\_MODE\_EQHIST: Internal automatic calculation of histogram statistics |
| u32HistArray | Histogram statistics of external input |
| u32HistMem | Histogram equalization auxiliary memory needs to be allocated at least 256 \* Memory of sizeof(RK\_U32) . |

### IVE\_CCL\_CTRL\_S

Define image connected area control parameters

"definition"

typedef struct rkIVE\_CCL\_CTRL\_S { IVE\_CCL\_MODE\_E enMode; RK\_U16 u16InitAreaThr;

RK\_U16 u16Step;

[IVE\_MEM\_INFO\_S](#_bookmark91) **stMem;**

**} IVE\_CCL\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | IVE\_CCL\_MODE\_4C: Four-connected mode  IVE\_CCL\_MODE\_8C: Eight-connected mode |
| u16InitAreaThr | Initial area threshold.  Value range: [0, 65535] . Reference value: 4 . |
| u16Step | The increment step of the area threshold. Value range: [1,65535] .  Reference value: 2 . |
| stMem | Auxiliary memory required for connected area calculations needs to be allocated at least as much memory as the input image buffer size |

### IVE\_CANNY\_EDGE\_CTRL\_S

Define image CANNY Edge detection control parameters

"definition"

typedef struct rkIVE\_CANNY\_HYS\_EDGE\_CTRL\_S {

[IVE\_MEM\_INFO\_S](#_bookmark91) stMem; RK\_U16 u16LowThr; RK\_U16 u16HighThr; RK\_S8 as8Mask[25];

**} IVE\_CANNY\_EDGE\_CTRL\_S;**

|  |  |
| --- | --- |
| stMem | Auxiliary memory, allocated at least one-quarter the size of the source image. |
| u16LowThr | Low threshold.  Value range: [0,255] . |
| u16HighThr | High threshold.  Value range: [u16LowThr,255] . |
| as8Mask[25] | Parameter template used to compute gradients. |

### IVE\_LBP\_CTRL\_S

Defining LBP Feature control parameters

"definition"

"illustrate"

typedef struct rkIVE\_LBP\_CTRL\_S { IVE\_LBP\_CMP\_MODE\_E enMode; IVE\_8BIT\_U un8BitThr;

**} IVE\_LBP\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | LBP comparison mode:  IVE\_LBP\_CMP\_MODE\_NORMAL : LBP Simple comparison mode  IVE\_LBP\_CMP\_MODE\_ABS : LBP Absolute value comparison mode |
| un8BitThr | LBP comparison threshold:  IVE\_LBP\_CMP\_MODE\_NORMAL The value range is: [-128,127] .  IVE\_LBP\_CMP\_MODE\_ABS The value range is [0,255] . |

### IVE\_GMM\_CTRL\_S

Defining GMM Background subtraction control parameters

"definition"

typedef struct rkIVE\_GMM\_CTRL\_S { RK\_U8 u8PicFormat;

RK\_U8 u8FirstFrameFlag; RK\_U8 u8EnBgOut; RK\_U8 u8MaxModelNum;

ive\_u8q2 u8q2WeightInitVal; ive\_u8q2 u8q2WeightAddFactor; ive\_u8q2 u8q2WeightReduFactor; ive\_u8q2 u8q2WeightThr;

RK\_U8 u8VarThreshGen;ive\_u8q2 u8q2BgRatio;

ive\_u10q0 u10q0InitVar; ive\_u10q0 u10q0MinVar; ive\_u10q0 u10q0MaxVar; RK\_U8 u8VarThr;

**} IVE\_GMM\_CTRL\_S;**

"illustrate"

|  |  |
| --- | --- |
| u8FirstFrameFlag | The first frame image is set to 1 , and subsequent calls are set to 0 |
| u8PicFormat | Input image format : 0: U8C1  1: U8C3 |
| u8EnBgOut | Output background control : 0: Do not output background  1 : Output background |
| u8MaxModelNum | Number of models.  Value range: {1,5} . |
| u8q2WeightInitVal | Initial value of model weight  Value range: {1,1023} . Reference value: 16 |
| u8q2WeightAddFactor | Model weight increase coefficient  Value range: {1,1023} . Reference value: 4 |
| u8q2WeightReduFactor | Model weight reduction coefficient  Value range: {1,1023} . Reference value: 1016 |
| u8q2WeightThr | Model destruction weight threshold  Value range: {1,1023} . Reference value: 4 |
| u8VarThreshGen | Foreground-background variance threshold  Reference value: 9 |
| u8q2BgRatio | Background calculation ratio setting |

u10q0InitVar

模型方差初始值

取值范围：{1,1023}。参考取值：225

u10q0MinVar

模型方差最小值

取值范围：{1,1023}。参考取值：200

u10q0MaxVar

模型方差最大值

取值范围：{1,1023}。参考取值：512

u8VarThr

模型方差阈值

取值范围：{1,1023}。参考取值：712

### IVE\_GMM2\_CTRL\_S

Defining GMM2 Background subtraction control parameters

"definition"

typedef struct rkIVE\_GMM2\_CTRL\_S { RK\_U8 u8PicFormat;

RK\_U8 u8FirstFrameFlag; RK\_U8 u8EnBgOut; RK\_U8 u8MaxModelNum; RK\_U8 u8UseVarFactor;

RK\_U8 u8GlobalLearningRateMode; RK\_U8 u8UpdateVar;

ive\_u8q2 u8q2WeightInitVal; ive\_u8q2 u8q2WeightAddFactor;

ive\_u8q2 u8q2WeightReduFactor; ive\_u8q2 u8q2WeightThr;

RK\_U8 u8VarThreshGen;ive\_u8q2 u8q2BgRatio;

ive\_u10q0 u10q0InitVar; ive\_u10q0 u10q0MinVar; ive\_u10q0 u10q0MaxVar; RK\_U8 u8VarThr;

**} IVE\_GMM2\_CTRL\_S;**

"illustrate"

|  |  |
| --- | --- |
| u8FirstFrameFlag | The first frame image is set to 1 , and subsequent calls are set to 0 |
| u8PicFormat | Input image format : 0: U8C1  1: U8C3 |
| u8EnBgOut | Output background control : 0: Do not output background  1 : Output background |
| u8MaxModelNum | Number of models.  Value range: {1,5} . |
| u8UseVarFactor | Pixel-level model update rate control  0 : Disable  1 : Enable |
| u8GlobalLearningRateMode | Global learning rate mode control  0 : Disable  1 : Enable |
| u8UpdateVar | Model variance update control  0 : Do not update  1 : Update |
| u8q2WeightInitVal | Initial value of model weight  Value range: {1,1023} . Reference value: 16 |
| u8q2WeightAddFactor | Model weight increase coefficient  Value range: {1,1023} . Reference value: 4 |

u8q2WeightReduFactor

模型权重减小系数

取值范围：{1,1023}。参考取值：1016

u8q2WeightThr

模型销毁权重阈值

取值范围：{1,1023}。参考取值：4

u8VarThreshGen

前景背景方差阈值

参考取值：9

u8q2BgRatio

背景计算比例设置

取值范围：{1,1023}。参考取值：712

u10q0InitVar

模型方差初始值

取值范围：{1,1023}。参考取值：225

u10q0MinVar

模型方差最小值

取值范围：{1,1023}。参考取值：200

u10q0MaxVar

模型方差最大值

取值范围：{1,1023}。参考取值：512

u8VarThr

模型方差阈值

### IVE\_LK\_OPTICAL\_FLOW\_CTRL\_S

the control parameters of the optical flow method (building a pyramid internally)

"definition"

"illustrate"

typedef struct rkIVE\_LK\_OPTICAL\_FLOW\_CTRL\_S { RK\_U16 u16PtsNum;

IVE\_U0Q8 u0q8MinEigThr; RK\_U8 u8IterCnt; IVE\_U0Q11 u0q11Eps;

**} IVE\_LK\_OPTICAL\_FLOW\_CTRL\_S;**

|  |  |
| --- | --- |
| u16PtsNum | Number of tracking points  Value range: [1, 200] |
| u0q8MinEigThr | Minimum eigenvalue threshold.  Value range: [1,255] . |
| u8IterCnt | Maximum number of iterations.  Value range: [1,20] . |
| u0q11Eps | Iterative convergence condition: dx^2 + dy^2 < u0q11Epsilon . Value range: [1, 4095] .  Reference value: 32 . |

### IVE\_LK\_OPTICAL\_FLOW\_PYR\_CTRL\_S

the control parameters of the optical flow method (external pyramid building)

"definition"

typedef struct rkIVE\_LK\_OPTICAL\_FLOW\_PYR\_CTRL\_S { IVE\_LK\_OPTICAL\_FLOW\_PYR\_OUT\_MODE\_E enOutMode; RK\_BOOL bUseInitFlow;

RK\_U16 u16PtsNum; RK\_U8 u8MaxLevel; IVE\_U0Q8 u0q8MinEigThr;

"illustrate"

RK\_U8 u8IterCnt;

IVE\_U0Q11 u0q11Eps;

**} IVE\_LK\_OPTICAL\_FLOW\_PYR\_CTRL\_S;**

|  |  |
| --- | --- |
| enOutMode | Output mode control of pstStatus and pstErr . |
| bUseInitFlow | Whether to use the initial optical flow calculation ( pstNextPts Initialization required ):  RK\_TRUE Indicates the use of initial optical flow, RK\_FALSE Indicates that the initial optical flow is not applicable. |
| u16PtsNum | Number of tracking points  Value range: [1, 500] |
| u8MaxLevel | u8MaxLevel+1 It is related to the number of pyramid layers.  Value range: [0, 3] , corresponding to the number of pyramid layers [1, 4] . Reference value: 2 . |
| u0q8MinEigThr | Minimum eigenvalue threshold.  Value range: [1,255] . |
| u8IterCnt | Maximum number of iterations.  Value range: [1,20] . |
| u0q11Eps | Iterative convergence condition: dx^2 + dy^2 < u0q11Epsilon . Value range: [1, 4095] .  Reference value: 32 . |

### IVE\_ST\_CANDI\_CORNER\_CTRL\_S;

Define the first step control parameters for image corner detection

"definition"

typedef struct rkIVE\_ST\_CANDI\_CORNER\_CTRL\_S {

"illustrate"

RK\_U8 u0q8QualityLevel;

**} IVE\_ST\_CANDI\_CORNER\_CTRL\_S;**

|  |  |
| --- | --- |
| u0q8QualityLevel​ | ShiTomasi corner quality control parameters, corner response value is less than "u0q8QualityLevel \* The point with the maximum corner point response value will be directly identified as a non- corner point.  Value range: [1,255] .  Reference value: 25 |

### IVE\_ST\_CORNER\_CTRL\_S;

Define the second step control parameters of image corner detection

"definition"

typedef struct rkIVE\_ST\_CORNER\_CTRL\_S {

[IVE\_MEM\_INFO\_S](#_bookmark91) **stMem; RK\_U16 u16MaxCornerNum; RK\_U16 u16MinDist;**

**} IVE\_ST\_CORNER\_CTRL\_S;**

|  |  |
| --- | --- |
| stMem | ShiTomasi corner quality control parameters, corner response value is less than "u0q8QualityLevel \* The point with the maximum corner point response value will be directly identified as a non- corner point.  Value range: [1,255] .  Reference value: 25 |
| u16MaxCornerN um | Maximum number of corner points.  Value range: [1,200] . |
| u16MinDist | Minimum distance between adjacent corner points. Value range: [1,65535] .  Reference value: 10 . |

### IVE\_MATCH\_BG\_MODEL\_CTRL\_S

Definition based on CODEBOOK Background subtraction model training control parameters

"definition"

typedef struct rkIVE\_MATCH\_BG\_MODEL\_CTRL\_S { RK\_U8 u8CodeWordNum;

RK\_U32 u32CurFrmNum; RK\_U8 u8TrainingTimeThr; RK\_U8 u8DiffMaxThr; RK\_U8 u8DiffMinThr;

**} IVE\_MATCH\_BG\_MODEL\_CTRL\_S;**

|  |  |
| --- | --- |
| u8CodeWordNum | Codeword Number. |
| u32CurFrmNum | Current frame ID |
| u8TrainingTimeThr | Training codebook Frame rate setting |
| u8DiffMaxThr | Training codebook Upper limit of pixel value |
| u8DiffMinThr | Training codebook Pixel value lower limit |

### IVE\_UPDATE\_BG\_MODEL\_CTRL\_S

Definition based on CODEBOOK Background subtraction model update control parameters

"definition"

typedef struct rkIVE\_UPDATE\_BG\_MODEL\_CTRL\_S { RK\_U8 u8CodeWordNum;

RK\_U32 u32CurFrmNum; RK\_U8 u8TimeThr; RK\_U8 u8DiffMaxThr; RK\_U8 u8DiffMinThr; RK\_U8 u8FastLearnRate; RK\_U8 u8Alpha;

**} IVE\_UPDATE\_BG\_MODEL\_CTRL\_S;**

|  |  |
| --- | --- |
| u8CodeWordNum | Codeword Number. |
| u32CurFrmNum | Current frame ID |
| u8TimeThr | Update the codebook Frame rate setting |
| u8DiffMaxThr | Update the codebook Upper limit of pixel value |
| u8DiffMinThr | Update the codebook Pixel value lower limit |
| u8FastLearnRate | Update rate  Value range: [1, 255] . Reference value: 16 . |
| u8Alpha | Codebook Update pixel value range ratio |

### IVE\_SAD\_CTRL\_S

Defining SAD Control parameters

"definition"

typedef struct rkIVE\_SAD\_CTRL\_S { IVE\_SAD\_MODE\_E enMode; IVE\_SAD\_OUT\_MODE\_E enOutMode; IVE\_SAD\_OUT\_BITS\_E enOutBits; RK\_U16 u16Thr;

RK\_U8 u8MinVal; RK\_U8 u8MaxVal;

**} IVE\_SAD\_CTRL\_S;**

|  |  |
| --- | --- |
| enMode | SAD Calculation mode. |
| enOutMode | SAD Output control mode. |
| enOutBits | SAD Output bit number : 0: 8 bits  1: 16bit |
| u16Thr | The calculated SAD The threshold value for thresholding the image. The value range depends on enMode : 1 , IVE\_SAD\_OUT\_CTRL\_8BIT\_BOTH ,  Values: [0, 255]  2. IVE\_SAD\_OUT\_CTRL\_16BIT\_BOTH​ and  IVE\_SAD\_OUT\_CTRL\_THRESH ,  Values: [0, 65535] |
| u8MinVal | thresholding does not exceed u16Thr . |
| u8MaxVal | The value when thresholding exceeds u16Thr . |

### IVE\_WARP\_AFFINE\_CTRL\_S

Define affine transformation control parameters

"definition"

typedef struct rkIVE\_WARP\_AFFINE\_CTRL\_S {

[IVE\_MEM\_INFO\_S](#_bookmark91) **stMem; RK\_FLOAT stAffineMat[6];**

**} IVE\_WARP\_AFFINE\_CTRL\_S;**

|  |  |
| --- | --- |
| stMem | Affine transformation auxiliary memory requires at least 5 times the cache space |
| stAffineMat[6]; | Affine transformation matrix |

### IVE\_PYRAMID\_CTRL\_S

Define the control parameters for generating image pyramids

"definition"

"illustrate"

typedef struct rkIVE\_PYRAMID\_CTRL\_S { [IVE\_MEM\_INFO\_S](#_bookmark91) stPyramidMem; RK\_U8 level;

**} IVE\_PYRAMID\_CTRL\_S;**

|  |  |
| --- | --- |
| stPyramidMem | Auxiliary memory required to generate the image pyramid, through RK\_MPI\_IVE\_Pyramid\_GetSize  Get the size of the space that needs to be opened. |
| level | Pyramid image layers |