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

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.

Rockchip Electronics Co., Ltd.

Rockchip Electronics Co., Ltd.

Address: Software Park A, Tongpan Road,

Fuzhou, Fujian District 18 Website : www.rock-chips.com

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

Customer Service Fax: +86-591-83951833

Customer Service Email: fae@rock-chips.com

Preface

Target audience

- Software Development Engineer
- Technical Support Engineer

Applicable platforms

- RV1103
- RV1106

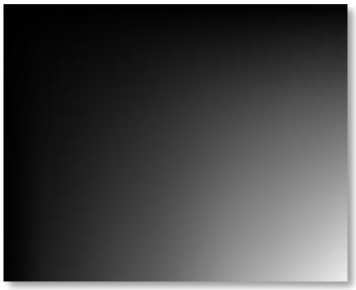
Version History

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

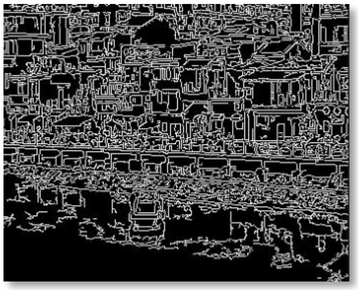
Table of contents

1. 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 ...

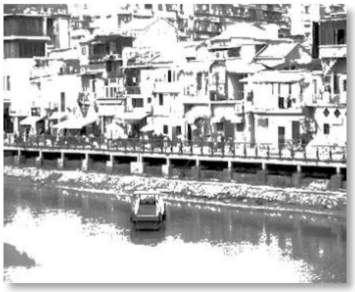


Integral
used as a quick and effective way of calculating the sum of pixel value



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

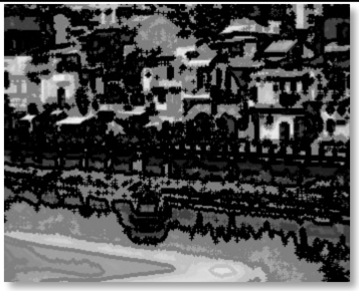
Filter
Filter image with a 5x5 kernel

Map
map one set of pixel values to another

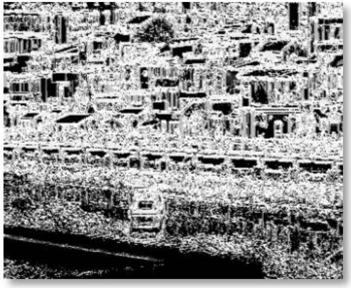
Min Filter
erodes shapes on the image

Median Filter
makes the target pixel luminosity equal to the mean value in the running window

Max Filter
extends object boundaries on the image



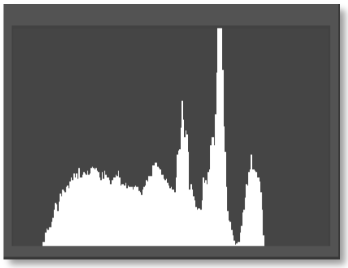
Erode
removes pixels on object boundaries



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



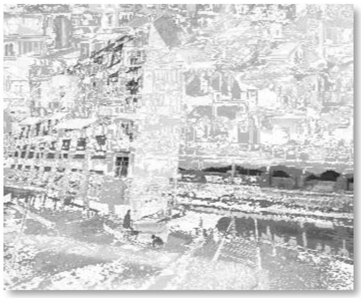
Histogram
acts as a graphical representation of the luminance distribution

Sub- Abs
subtract two images and calculate the absolute value

Sub-Shift
subtract two images and right shift 1 bit of the result

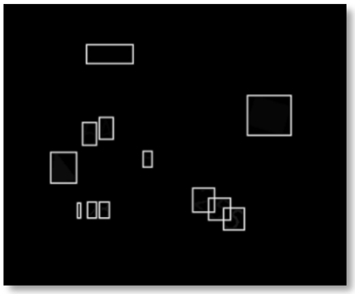
Add
combine two images with different weight

And
bitwise and of two images

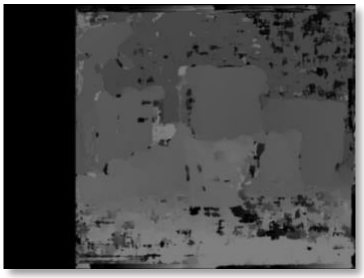


Or
bitwise or of two images

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

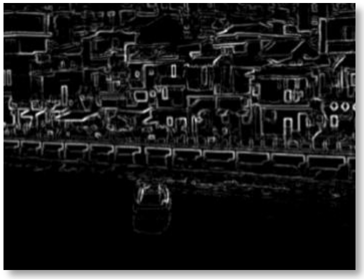
normalized cross-correlation, can be used to determine how to register or align the images

LK Optical Flow

optical flow estimation

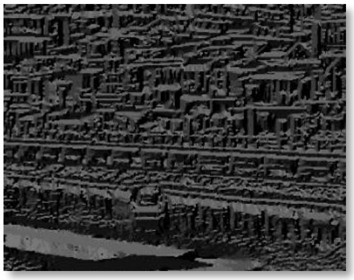
GMM & BGM

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

2. Getting Started

2.1. Environmental Description

2.1.1. 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 .

2.1.2. Input and output buffering

RKIVE The input and output buffer types required by the functional interface include IVE_MEM_INFO_S , IVE_DATA_S , and IVE_DST_IMAGE_S . 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.

2.2. 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 , 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 .

- 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 . 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 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.

2.3. Reference Examples

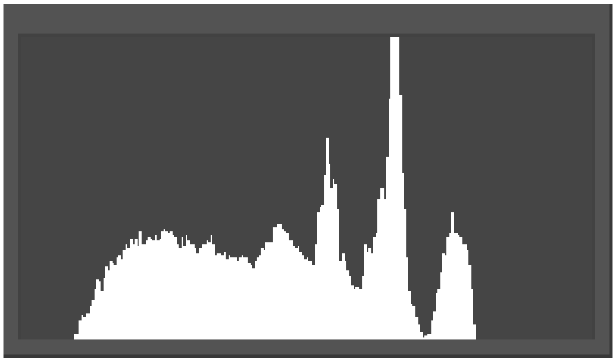


Figure 2-1

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 ;
}
```

3. **Proc** Debug information

3.1. 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!
```

3.2. **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
```

3.3. Error Code

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

4. Image format diagram

YCbCr 4:0:0 format

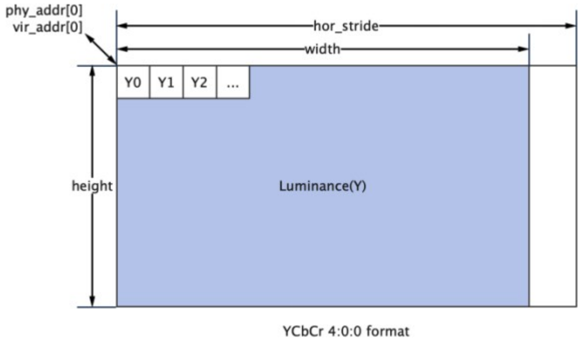


Figure 4-1

YUV420SP format

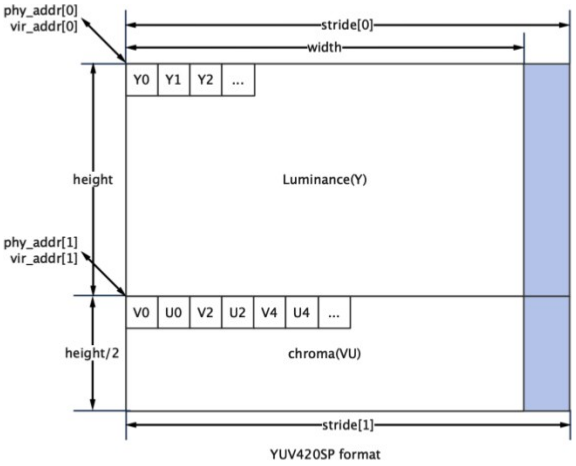


Figure 4-2

YUV420P format

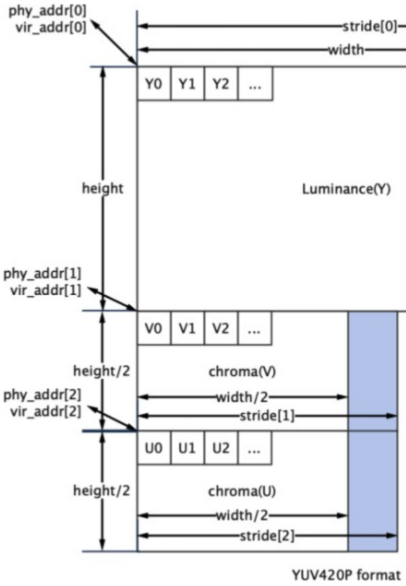


Figure 4-3

YUV422SP format

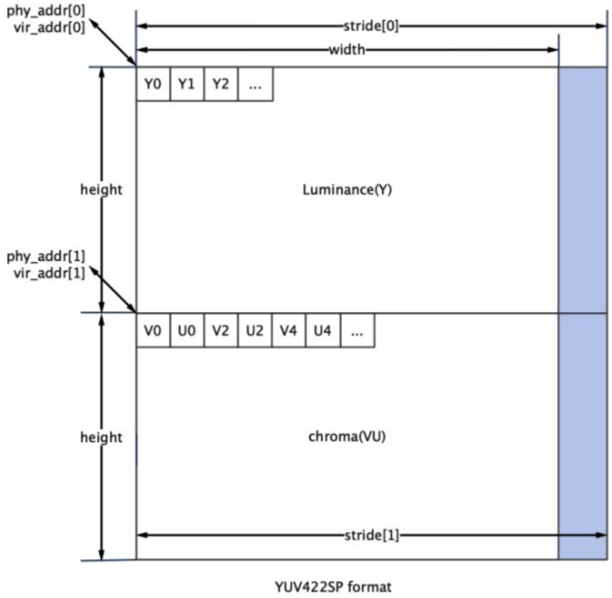


Figure 4- 4

YUV422P format

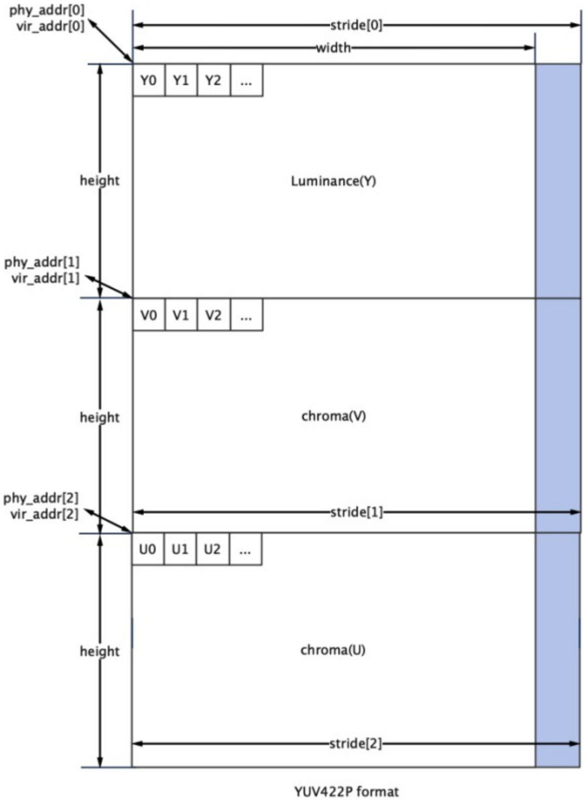


Figure 4-5

U8C1 format

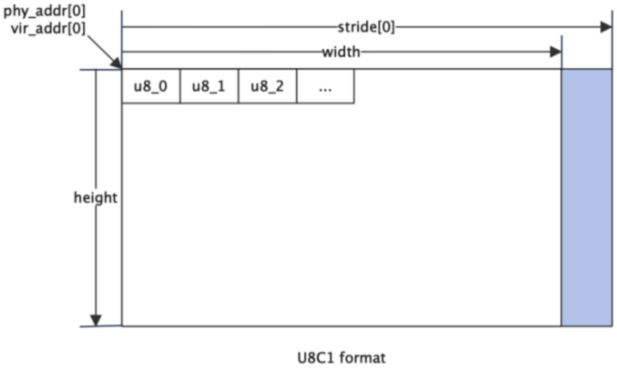


Figure 4-6

U8C3 planar format

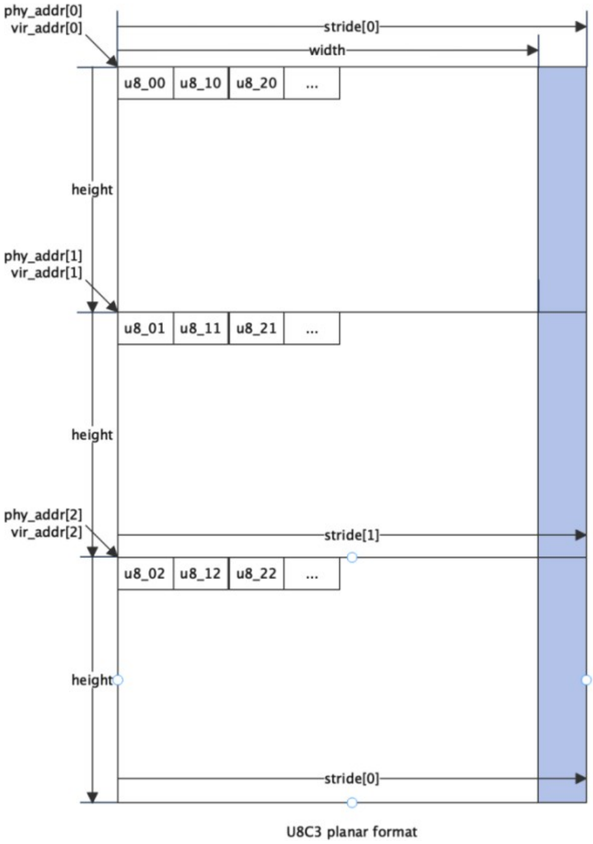


Figure 4-7

U8C3 package format

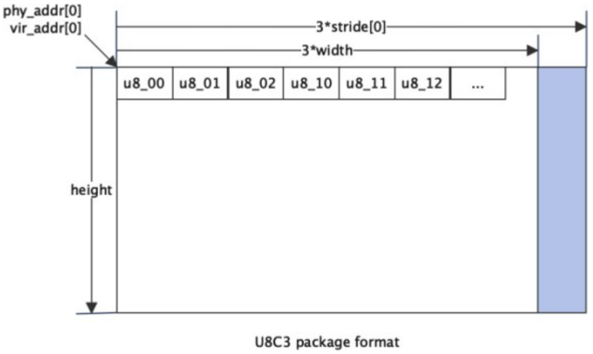


Figure 4-8

5. **API** refer to

RKIVE Provides the following functional interfaces:

- RK_MPI_IVE_Init：Complete IVE Context creation and necessary initialization work.
- RK_MPI_IVE_Deinit：Complete IVE The context is destroyed.
- RK_MPI_IVE_CvtImageToData : A single planar Image type is converted to type.
- RK_MPI_IVE_CvtDataToImage : Converts a two-dimensional data type to a single planar Image type.
- RK_MPI_IVE_CvtImageToMemInfo： A single planar Converts the image type to a one-dimensional data type.
- RK_MPI_IVE_DMA： Direct memory access, supporting fast copy, interval copy, and memory filling.
- RK_MPI_IVE_Filter： 5X5 Template filtering tasks can achieve different filtering by configuring different template coefficients.
- RK_MPI_IVE_CSC：color space conversion, which can realize YUV , RGB , HSV Color space conversion.
- RK_MPI_IVE_Sobel：3X3 or 5X5 Template SOBEL-LIKE Gradient calculation.
- RK_MPI_IVE_MagAndAng： Calculate the amplitude argument.
- RK_MPI_IVE_Dilate： Image dilation, image morphological operation.
- RK_MPI_IVE_Erode： Image erosion, image morphological operations.
- RK_MPI_IVE_Add : Weighted addition operation of two grayscale images.
- RK_MPI_IVE_And : Perform AND operation on two grayscale images.
- RK_MPI_IVE_Sub : Subtract two grayscale images.
- RK_MPI_IVE_Or : OR operation between two grayscale images.
- RK_MPI_IVE_Xor : Perform the XOR operation on two grayscale images.
- RK_MPI_IVE_Integ： Image integral map calculation.
- RK_MPI_IVE_Hist： Image histogram calculation.
- RK_MPI_IVE_Thresh : Image 8 Grayscale image binarization operation.
- RK_MPI_IVE_Thresh_U 16 : 16 bit data to 8 Bit data threshold binarization operation.
- RK_MPI_IVE_Thresh_S16 : 16 with sign bit bit data to 8 Threshold binarization operation for bit data.
- RK_MPI_IVE_16BitTo8Bit : 16 bit data to 8 Linear conversion of bit data.
- RK_MPI_IVE_8BitTo8Bit : 8 bit data to 8 Linear conversion of bit data.

- RK_MPI_IVE_OrdStatFilter： Image median filtering, minimum filtering, and maximum filtering.
- RK_MPI_IVE_Map : Image pixel value mapping is performed according to the lookup table.
- RK_MPI_IVE_EqualizeHist： Image histogram equalization.

- RK_MPI_IVE_NCC： Normalized cross-correlation coefficient between two grayscale images of the same resolution.
- RK_MPI_IVE_CCL：Connected component labeling for binary images.
- RK_MPI_IVE_GMM： Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG .
- RK_MPI_IVE_GMM2： Create a Gaussian mixture background model and perform foreground-background separation operations, refer to OPENCV MOG2 .
- RK_MPI_IVE_CannyEdge： Extract edge information from grayscale images.
- RK_MPI_IVE_LBP：Calculate image LBP feature.
- RK_MPI_IVE_NormGrad： Image normalized gradient calculation, all gradient components are normalized to S8 .
- RK_MPI_IVE_LKOpticalFlowPyr： LK Optical flow calculation (external pyramid building) .
- RK_MPI_IVE_LKOpticalFlow： LK Optical flow calculation (building a pyramid internally) .
- 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_MPI_IVE_STCorner：The second step of image corner detection is to sort the candidate corner points according to the rules.
- RK_MPI_IVE_MatchBgModel： Based on CODEBOOK The first step of background subtraction is background model training.
- RK_MPI_IVE_UpdateBgModel： 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： Initialize affine transformation auxiliary memory.
- RK_MPI_IVE_Warp_Affine： Perform image affine transformation tasks.
- RK_MPI_IVE_Pyramid_GetSize： Gets the auxiliary memory size required to generate the image pyramid.
- RK_MPI_IVE_Pyramid_Create： Executes the task of creating an image pyramid.
- RK_MPI_IVE_Query： 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 *image,
                                   IVE_DATA_S *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.	enter

	Cannot be empty.	
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.	enter

	Cannot be empty.	
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 *pstSrc,

IVE_DST_DATA_S *pstDst,

IVE_DMA_CTRL_S *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	1 byte	1x1~2047x2047
pstDst	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" :

- 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 *pstSrc,
IVE_DST_IMAGE_S *pstDst,
IVE_FILTER_CTRL_S *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 :

$$I_{out}(x, y) = \{ \sum_{-2 \le j \le 2} \sum_{-2 \le i \le 2} I(x+i, y+j) * coef(i, j) \} \gg norm$$

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 *pstSrc,
                             IVE_DST_IMAGE_S *pstDst,
                             IVE_CSC_CTRL_S *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 *pstSrc,
                         IVE_DST_IMAGE_S * pstDstH,
                         IVE_SRC_IMAGE_S * pstDstV ,
                         IVE_SOBEL_CTRL_S *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:

$$H_{out}(x, y) = \{ \sum_{-2 \le j \le 2} \sum_{-2 \le i \le 2} I(x+i, y+j) * coef(i, j) \} \gg norm$$

$$V_{out}(x, y) = \{ \sum_{-2 \le j \le 2} \sum_{-2 \le i \le 2} I(x+i, y+j) * coef(i, j) \} \gg norm$$

RK_MPI_IVE_MagAndAng

Calculate the amplitude and angle

```
RK_S32 RK_MPI_IVE_MagAndAng (IVE_HANDLE *pHandle,
                             IVE_SRC_IMAGE_S *pstSrc,
                             IVE_DST_IMAGE_S *pstDstMag,
                             IVE_DST_IMAGE_S *pstDstAng,
                             IVE_MAG_AND_ANG_CTRL_S *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 :

$$H_{out} = \sum_{-2\leq j\leq 2} \sum_{-2\leq i\leq 2} I\,(\,x+i,\, y+j\,) * coef(i,\, j)$$

$$V_{out} = \sum_{-2\leq j\leq 2} \sum_{-2\leq i\leq 2} I\,(\,x+i,\, y+j\,) * coef(j,\, i)$$

$$Mag\,(\,x,\, y\,) = abs(H_{out}(\,\mathrm{igh},\, y\,)\,) + abs(V_{out}(\,\mathrm{igh},\, y\,)\,)$$

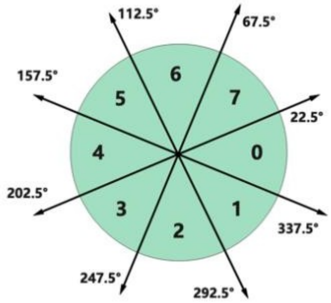
Where $I\,(\,x\,,\,y\,)$ is the input source image, $coef(mask)$ is the 5x5 template coefficient for calculating the gradient , H_{out} is the horizontal image gradient, and V_{out} is the vertical image gradient. $Mag\,(\,x\,,\,y\,)$ is the image gradient amplitude.

- The formula for calculating the argument is as follows :

$$H_{out} = \sum_{-2\leq j\leq 2} \sum_{-2\leq i\leq 2} I\,(\,x+i,\, y+j\,) * coef(i,\, j)$$

$$V_{out} = \sum_{-2\leq j\leq 2} \sum_{-2\leq i\leq 2} I\,(\,x+i,\, y+j\,) * coef(j,\, i)$$

$$\theta\,(\,x,\, y\,) = arctan\,(\frac{V_{ou\ t}}{H_{out}})$$



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 *pstSrc,
                           IVE_DST_IMAGE_S *pstDst,
                           IVE_DILATE_CTRL_S *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 *pstSrc,
                         IVE_DST_IMAGE_S *pstDst,
                         IVE_ERODE_CTRL_S *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:

$$I_{out}(x, y) = (I(x-2, y-2)\&coef(-2,-2)) \& (I(x-1, y-2)\&coef(-1,-2)) \& \ldots \& (I(x+2, y+2)\&coef(2,2))$$

RK_MPI_IVE_Add

Weighted addition operation of two grayscale images

```
RK_S32 RK_MPI_IVE_Add (IVE_HANDLE *pHandle,
                        IVE_SRC_IMAGE_S *pstSrc1,
                        IVE_SRC_IMAGE_S *pstSrc2,
                        IVE_DST_IMAGE_S *pstDst,
                        IVE_ADD_CTRL_S *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:

$$I_{out}(i, j) = ((x * I_{src1}(i, j) + y * I_{src2}(i, j)) \gg 16)\&0xff$$

RK_MPI_IVE_And

AND operation between two grayscale images

```
RK_S32 RK_MPI_IVE_And (IVE_HANDLE *pHandle,
                       IVE_SRC_IMAGE_S *pstSrc1,
                       IVE_SRC_IMAGE_S *pstSrc2,
                       IVE_DST_IMAGE_S *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 enter in time.

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 *pstSrc1,

IVE_SRC_IMAGE_S *pstSrc2,

IVE_DST_IMAGE_S *pstDst,

IVE_SUB_CTRL_S *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 *pstSrc1,
IVE_SRC_IMAGE_S *pstSrc2,
IVE_DST_IMAGE_S *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:

$$I_{out}(i, j) = I_{src1}(i, j) \mid I_{src2}(i, j)$$

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 *pstSrc1,
                       IVE_SRC_IMAGE_S *pstSrc2,
                       IVE_DST_IMAGE_S *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	librve.a , librve.so

```
Files
```

"Notice" :

- Image difference calculation formula:

$$I_{out}(i, j) = I_{src1}(i, j)\wedge I_{src2}(i, j)$$

RK_MPI_IVE_Integ

Image integral map calculation

```
RK_S32 RK_MPI_IVE_Integ (IVE_HANDLE *pHandle,
                         IVE_SRC_IMAGE_S *pstSrc,
                         IVE_DST_IMAGE_S *pstDst,
                         IVE_INTEG_CTRL_S *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:

$$I_{sum}(x, y) = \sum_{i\geq 0}^{i\leq x} \sum_{j\geq 0}^{j\leq y} I(i, j)$$

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:

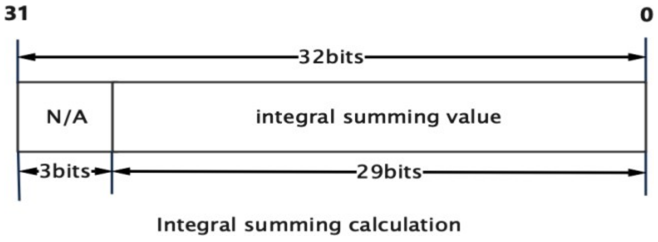


Figure 5-2

- Square sum integral graph (U64C1): mode IVE_INTEG_OUT_CTRL_SQSUM , the calculation formula is as follows :

$$I_{sq}(x, y) = \sum_{i\geq 0}^{i\leq x} \sum_{j\geq 0}^{j\leq y}(I(i, j) * I(i, j))$$

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:

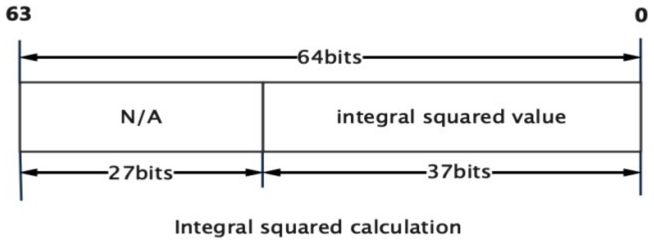


Figure 5-3

- Sum integral and square sum integral combined (U64C1): mode IVE_INTEG_OUT_CTRL_COMBINE , the calculation formula is as follows:

$$I_{OUT}(X, Y) = (I_{SQ}(X, Y) \ll 28)) \mid (I_{SUM}(X, Y)\&0XFFFFFF)$$

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

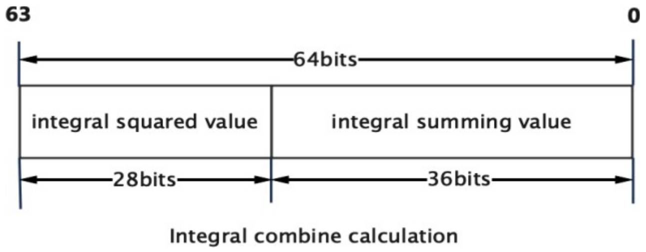


Figure 5- 4

- 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 *pstSrc,

IVE_DST_MEM_INFO_S *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:

$$hist(x) = \sum_i \sum_j ((I(i,j) == x)?\ 1:0) \qquad x = 0..255$$

RK_MPI_IVE_Thresh

Image 8 Grayscale image binarization operation

```
RK_S32 RK_MPI_IVE_Thresh (IVE_HANDLE *pHandle,
                          IVE_SRC_IMAGE_S *pstSrc,
                          IVE_DST_IMAGE_S *pstDst,
                          IVE_THRESH_U8_CTRL_S *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:

$$I_{out}(x, y) = \begin{cases} m \quad nVal & (I(x, y) \le lowThr) \\ maighVal & (I(igh, y) > lowThr) \end{cases}$$

- TRUNC:

$$I_{out}(x, y) = \begin{cases} I(x, y) & (I(x, y) \le lowThr) \\ maighVal & (I(igh, y) > lowThr) \end{cases}$$

- MINVAL:

$$I_{out}(x, y) = \begin{cases} m \quad nVal & (I(x, y) \le lowThr) \\ I(x, y) & (I(x, y) > lowThr) \end{cases}$$

- MIN_MID_MAX:

$$I_{out}(x, y) = \begin{cases} minVal & (I(x, y) \le lowThr) \\ midVal & (lowThr < I(x, y) \le highThr) \\ maxVal & (I(x, y) > highThr) \end{cases}$$

- ORI_MID_MAX:

$$I_{out}(x, y) = \begin{cases} I(x, y) & (I(x, y) \le lowThr) \\ midVal & (lowThr < I(x, y) \le highThr) \\ maxVal & (I(x, y) > highThr) \end{cases}$$

- MIN_MIN_ORI:

$$I_{out}(x, y) = \begin{cases} minVal & (I(x, y) \leq lowThr) \\ midVal & (lowThr < I(x, y) \leq highThr) \\ I(x, y) & (I(x, y) > highThr) \end{cases}$$

- MIN_ORI_MAX:

$$I_{out}(igh, y) = \begin{cases} minVal & (I(x, y) \leq lowThr) \\ I(x, y) & (lowThr < I(x, y) \leq highThr) \\ maxVal & (I(x, y) > highThr) \end{cases}$$

- ORI_MID_ORI:

$$I_{out}(x, y) = \begin{cases} I(x, y) & (I(x, y) \leq lowThr) \\ midVal & (lowThr < I(x, y) \leq highThr) \\ I(x, y) & (I(x, y) > highThr) \end{cases}$$

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 *pstSrc,
                              IVE_DST_IMAGE_S *pstDst,
                              IVE_THRESH_U16_CTRL_S *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	enter

in time.

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 *pstSrc,
                       IVE_DST_IMAGE_S *pstDst,
                       IVE_THRESH_U16_CTRL_S *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 *pstSrc,
                       IVE_DST_IMAGE_S *pstDst,
                       IVE16BIT_TO_8BIT_CTRL_S *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
pst16BitTo8BitCtrl	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 *pstSrc,

IVE_DST_IMAGE_S *pstDst,

```
IVE_8BIT_TO_8BIT_CTRL_S *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
pst8BitTo8BitCtrl	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 *pstSrc,
                                 IVE_DST_IMAGE_S *pstDst,
                                 IVE_ORD_STAT_FILTER_CTRL_S *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 :

$$I_{out}(x, y) = med\mathrm{ade}an_{-1\le j\le 1}^{-1\le \quad \le 1} \{ I(x + i, y + j) \}$$

- IVE_ORD_STAT_FILTER_MODE_MAX :

$$I_{out}(x, y) = ma\mathrm{igh}_{-1\le j\le 1}^{-1\le \quad \le 1} \{ I(\mathrm{igh} + i, y + j) \}$$

- IVE_ORD_STAT_FILTER_MODE_MIN :

$$I_{out}(x, y) = m \quad n_{-1\le j\le 1}^{-1\le \quad \le 1} \{ I(x + i, y + j) \}$$

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 *pstSrc,
                       IVE_SRC_MEM_INFO_S *pstMap
                       IVE_DST_IMAGE_S *pstDst,
                       IVE_MAP_CTRL_S *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 *pstSrc,
                                IVE_DST_IMAGE_S *pstDst,
                                IVE_EQHIST_CTRL_S *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 *pstSrc1,
                       IVE_SRC_IMAGE_S *pstSrc2,
                       IVE_DST_MEM_INFO_S *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:

$$NCC(I_{src1}, I_{src2}) = \frac{\sum_{i=1}^{w} \sum_{j=1}^{h} (I_{src1}(i,j) * I_{src2}(i,j))}{\sqrt{\sum_{i=1}^{w} \sum_{j=1}^{h} (I^2_{src1}(i,j))} \sqrt{\sum_{i=1}^{w} \sum_{j=1}^{h} (I^2_{src2}(i,j))}}$$

RK_MPI_IVE_CCL

Connected component labeling for binary images.

RK_S32 **RK_MPI_IVE_CCL** (IVE_HANDLE *pHandle,

```
 IVE_SRC_IMAGE_S * pstSrcDst,

IVE_DST_MEM_INFO_S *pstBlob,

IVE_CCL_CTRL_S *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"：

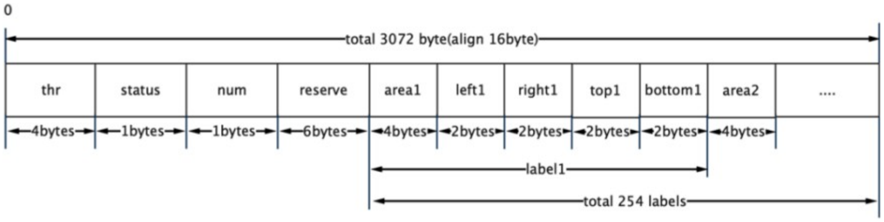
parameter	Data Type	Address alignment	Resolution
pstSrcDst	U8C1	1 byte	64x64~1984x2047
pstBlob	U8C1	16 bytes	64x64~1984x2047

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 :



CCL information in DDR

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 *pstSrc,
                       IVE_DST_IMAGE_S *pstFg,
                       IVE_DST_IMAGE_S *pstBg,
                       IVE_DST_IMAGE_S *pstMatchModelInfo,
                       IVE_MEM_INFO_S *pstModel,
                       IVE_GMM_CTRL_S *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	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 *pstSrc,
```

IVE_SRC_IMAGE_S *pstFactor,

IVE_DST_IMAGE_S *pstFg,

```
                    IVE_DST_IMAGE_S *pstBg,
                    IVE_DST_IMAGE_S *pstMatchModelInfo,
                    IVE_MEM_INFO_S *pstModel,
                    IVE_GMM2_CTRL_S *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	enter

time .

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	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:

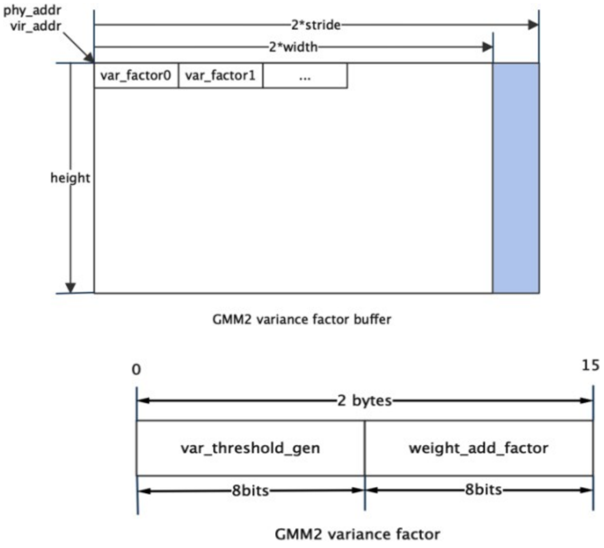


Figure 5-6

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

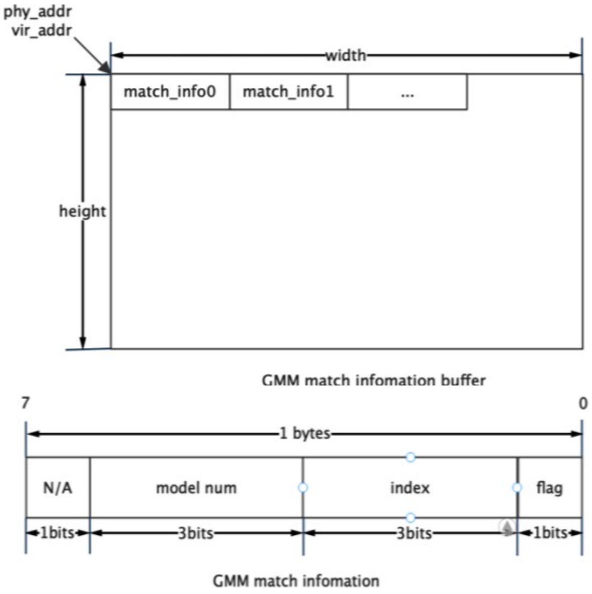
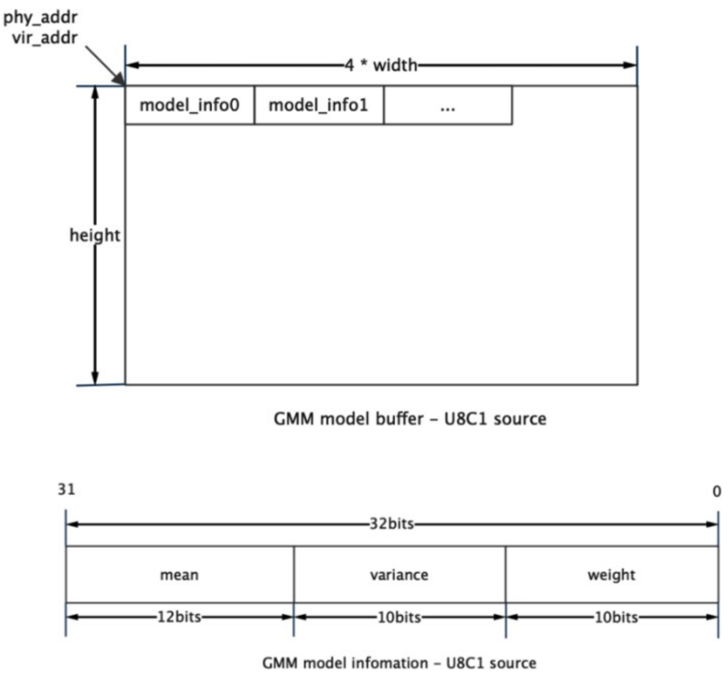


Figure 5-7

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

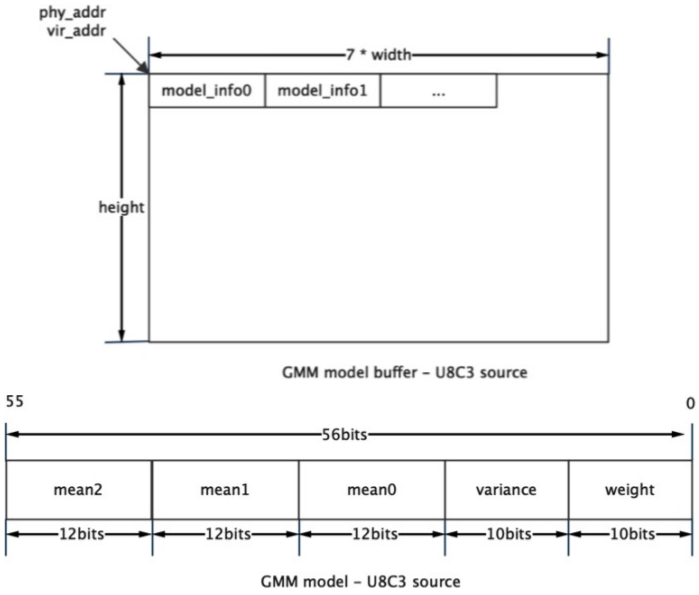


GMM model buffer – U8C1 source

GMM model infomation – U8C1 source

Figure 5-8

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



GMM model buffer – U8C3 source

GMM model – U8C3 source

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 *pstSrc,
                             IVE_DST_IMAGE_S *pstEdge,
                             IVE_DST_MEM_INFO_S *pstStack,
                             IVE_CANNY_EDGE_CTRL_S *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
pstCannyEdgeCtrl	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	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 *pstSrc,
                       IVE_DST_IMAGE_S *pstDst,
                       IVE_LBP_CTRL_S *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:

$((I_{\text{walk}} - I_c) \geq \text{falcon}hr) \ll (7 - \text{walk})$, where

$lbp(x, y) = \sum_{i=0}^{7}$ falcon$hr \in [-128,127]$ $(abs(I \ {}_{-} I\ c\ {}_{)} \geq \text{falcon}hr) \ll (7$

$lbp(x, y) = \sum_{i=0}^{7} - \text{walk})$, where falcon$hr \in [\ 0,255]$

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 *pstSrc,
                            IVE_DST_IMAGE_S *pstDstH,
                            IVE_DST_IMAGE_S *pstDstV,
                            IVE_DST_IMAGE_S *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 enter time.

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:

$$I_{out}(x, y) = \{ \sum_{-2 \le j \le 2} \sum_{-2 \le i \le 2} I(x+i, y+j) * coef(i, j) \} \gg norm$$

RK_MPI_IVE_LKOpticalFlowPyr

LK Optical flow calculation (external pyramid building) .

RK_S32 **RK_MPI_IVE_LKOpticalFlowPyr** (IVE_HANDLE *pHandle,

IVE_SRC_IMAGE_S astSrcPrevPyr[],

IVE_SRC_IMAGE_S astSrcNextPyr[],

```
IVE_SRC_MEM_INFO_S *pstPrevPts,
IVE_MEM_INFO_S *pstNextPts,
IVE_DST_MEM_INFO_S *pstStatus,
IVE_DST_MEM_INFO_S *pstErr,
IVE_LK_OPTICAL_FLOW_PYR_CTRL_S *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	Output

	similarity error estimation . Cannot be empty.	
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 *pstSrcPre,
IVE_SRC_IMAGE_S *pstSrcCur,
IVE_SRC_MEM_INFO_S *pstPoint,
IVE_SRC_MEM_INFO_S *pstMv,
IVE_LK_OPTICAL_FLOW_CTRL_S *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:

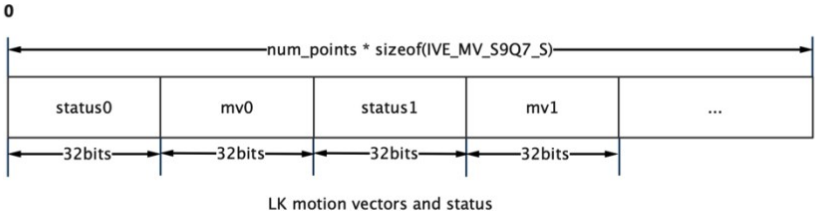

0
num_points * sizeof(IVE_MV_S9Q7_S)
status0
mv0
status1
mv1
...
32bits
32bits
32bits
32bits
LK motion vectors and status

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 *pstSrc,

IVE_DST_MEM_INFO_S *pstCandiCorner,

IVE_ST_CANDI_CORNER_CTRL_S *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 ;

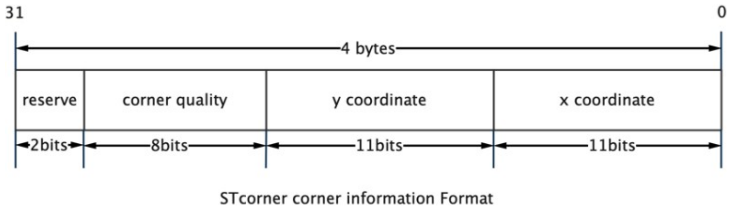


Figure 5-11

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 *pstSrc,
                            IVE_DST_MEM_INFO_S *pstCandiCorner,
                            IVE_DST_MEM_INFO_S *pstCorner,
IVE_ST_CORNER_CTRL_S *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:

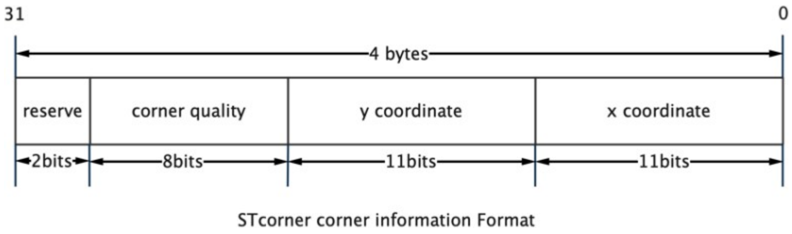


Figure 5-12

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 *pstCurImg,
 IVE_DATA_S *pstBgModel,

 IVE_MATCH_BG_MODEL_CTRL_S *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:

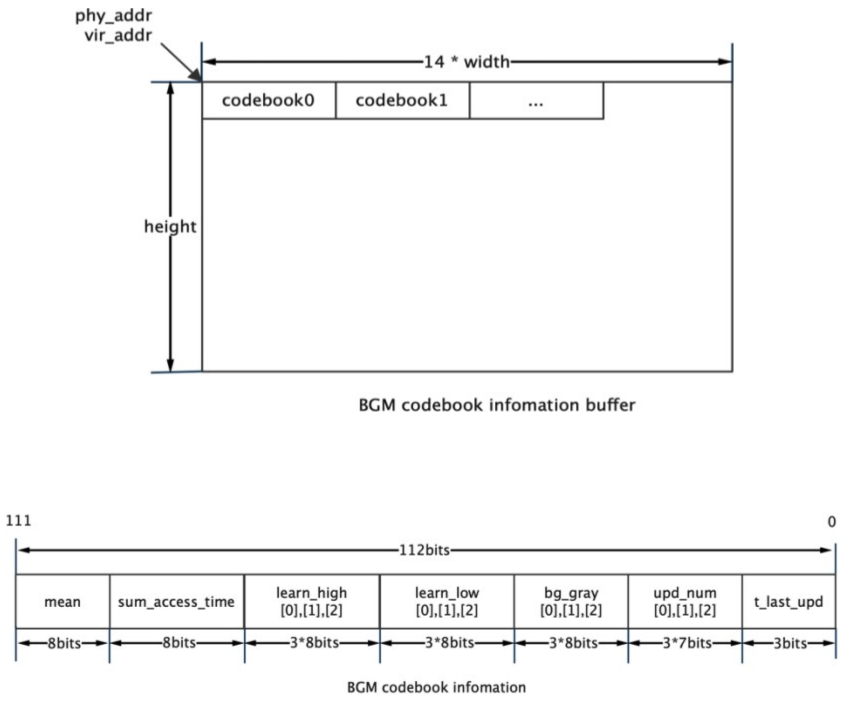


Figure 5-13

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 * pstCurImg,
                                 IVE_DATA_S *pstBgModel,
                                 _IVE_IMAGE_S *pstFgFlag,
                                 IVE_DST_IMAGE_S *pstBgImg,
                                 IVE_UPDATE_BG_MODEL_CTRL_S *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":

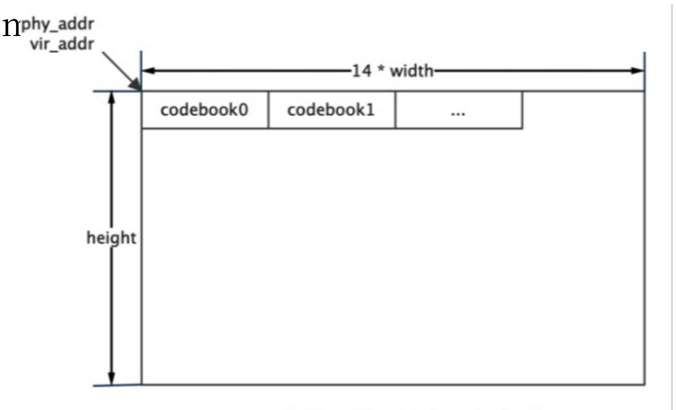
parameter	Data Type	Address alignment	Resolution
pstCurImg	U8C1	1 byte	1x1~2047x2047
pstBgModel	-	1 byte	-
pstFgFlag	U8C1	1 byte	1x1~2047x2047
pstBgImg	U8C1	1 byte	1x1~2047x2047

Quote :

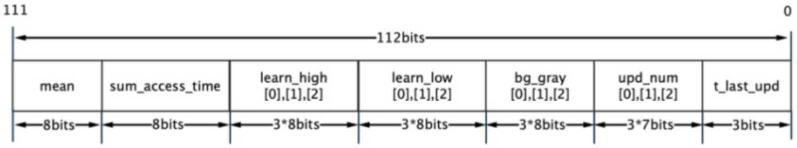
header files	rk_comm_ive.h , rk_ive.h , rk_mpi_ive.h
Library Files	librve.a , librve.so

"Notice" :

- pstBgModel Storage book Model, each model memory size is 14 bytes, whose format in men



BGM codebook infomation buffer



BGM codebook infomation

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 * pstSrc1,
                       IVE_SRC_IMAGE_S *pstSrc2,
                       IVE_DST_IMAGE_S *pstSad,
                       IVE_DST_IMAGE_S *pstThr,
                       IVE_SAD_CTRL_S *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	enter

time.

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:

$$\mathrm{Diff}(i, j) = |I_1(i, j) - I_2(i, j)|$$

$$SAD_{out}(x, y) = \sum_{n*x \le i < n*(x+1)\ \ n*y \le j < n*(y+1)} \mathrm{Diff}(i, j), \quad (x \ge 0, y \ge 0, stride = n)$$

$$THR_{out}(x, y) = \begin{cases} minVal & (SAD_{out}(x, y) \le Thr) \\ maxVal & (SAD_{out}(x, y) > Thr) \end{cases}$$

in,

n= 4 Corresponding to

RVE_SAD_MODE_MB_4X4 n= 8

Corresponding to RVE_SAD_MODE_M	B_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 *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 *pstSrc,
                               IVE_DST_IMAGE_S *pstDst,
                               IVE_WARP_AFFINE_CTRL_S *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:

$$\begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{01} \end{bmatrix}$$

in,

$\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}$ is the linear transformation correlation coefficient, and $\begin{bmatrix} b_{00} \\ b_{01} \end{bmatrix}$ is the translation correlation coefficient.

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 *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 *pstSrc,
IVE_DST_IMAGE_S pstPyramid[],
IVE_PYRAMID_CTRL_S *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	librve.a , librve.so

```
Files
```

"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:

$$width_{n+1} = \frac{(width_n + 1)}{2}$$

$$height_{n+1} = \frac{(height_n + 1)}{2}$$

- 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 .

6. 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;
```

```
typedef IVE_DATA_S IVE_SRC_DATA_S;
typedef IVE_DATA_S IVE_DST_DATA_S;
```

"illustrate"

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;

「说明」

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;
```

「说明」

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;

「说明」

「说明」

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;
```

「说明」

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.
u8YUV2RGBRange	YUV Convertto RGB Mode data range: 0: [16~235] 1: [0~255]
u8RGB2YUVRange	RGB Convertto YUV Mode data range: 0: [16~235] 1: [0~255]

IVE_SOBEL_CTRL_S

Defining SOBEL Control Information

"definition"

```
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;
```

"illustrate"

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"

```
typedef struct rkIVE_MAG_AND_ANG_CTRL_S
   { RK_S8 as8Mask[25];
   IVE_MEM_INFO_S stMem;
} IVE_MAG_AND_ANG_CTRL_S;
```

"illustrate"

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"

```
typedef struct rkIVE_DILATE_CTRL_S
   { RK_U8 au8Mask[25];
} IVE_DILATE_CTRL_S;
```

"illustrate"

au8Mask	5x5 Template coefficient

IVE_ERODE_CTRL_S

Define image corrosion control information

"definition"

```
typedef struct rkIVE_ERODE_CTRL_S
   { RK_U8 au8Mask[25];
} IVE_ERODE_CTRL_S;
```

"illustrate"

au8Mask	5x5 Template coefficient

IVE_ADD_CTRL_S

Define image weighting and control information

"definition"

```
typedef struct rkIVE_ADD_CTRL_S
   { RK_U16 u0q16X;
   RK_U16 u0q16Y;
} IVE_ADD_CTRL_S;
```

"illustrate"

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"

```
typedef struct rkIVE_SUB_CTRL_S
   { IVE_SUB_MODE_E enMode;
} IVE_SUB_CTRL_S;
```

"illustrate"

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 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;
```

「说明」

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;
```

「说明」

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"

```
typedef struct rkIVE_MAP_CTRL_S
   { IVE_MAP_MODE_E enMode;
} IVE_MAP_CTRL_S;
```

"illustrate"

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 u32HistMem;
} IVE_EQHIST_CTRL_S;
```

「说明」

「说明」

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 stMem;
} IVE_CCL_CTRL_S;
```

「说明」

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 stMem;
  RK_U16 u16LowThr;
  RK_U16 u16HighThr;
  RK_S8 as8Mask[25];
} IVE_CANNY_EDGE_CTRL_S;
```

「说明」

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"

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

"illustra
te"

enMode	LBP comparison mode: IVE_LBP_CMP_MODE_NORMAL : LBP Simple comparison mode IVE_LBP_CMP_MODE_ABS : LBP Absolute value comparison mode

「说

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_u8q
   2 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

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_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_u8q
    2 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"

```
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;
```

"illustrate"

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;

```
        RK_U8 u8IterCnt;
        IVE_U0Q11 u0q11Eps;
    } IVE_LK_OPTICAL_FLOW_PYR_CTRL_S;
```

"illustrate"

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 {
    RK_U8 u0q8QualityLevel;
} IVE_ST_CANDI_CORNER_CTRL_S;
```

"illustrate"

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 stMem;
    RK_U16 u16MaxCornerNum;
    RK_U16 u16MinDist;
} IVE_ST_CORNER_CTRL_S;
```

「说明」

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
u16MaxCornerNum	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;
```

「说明」

「说明」

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;
```

「说明」

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;
```

「说明」

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 stMem;
   RK_FLOAT stAffineMat[6];
} IVE_WARP_AFFINE_CTRL_S;
```

「说明」

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"

```
typedef struct rkIVE_PYRAMID_CTRL_S {
   IVE_MEM_INFO_S stPyramidMem;
   RK_U8 level;
} IVE_PYRAMID_CTRL_S;
```

"illustrate"

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