编译与运行 sample

sample\_venc(视频数据编码)

sample\_vdec(视频数据解码)

sample\_audio(音频相关)

sample\_snap(拍照)

sample\_dpu\_main(Depth Process Unit,深度图)

sample\_avs(Any View Stitching,全景拼接)

sample\_fisheye(鱼眼模式)

sample\_vio(视频输入输出)

sample\_vgs(Video Graphics Sub-System,视频图形子系统)

sample\_tde(Two Dimensional Engine,利用硬件为OSD和GUI提供快速的图形绘制功能)

sample\_hifb(Hisilicon Framebuffer,基于Linux FB 基本功能扩展了一些图形层控制功能)

sample\_awb\_calibration(自动白光平衡测量)

sample\_ive\_main(智能加速引擎)

sample\_dis(Digital Image Stabilization,数字稳像)

sample\_dsp\_main(DSP测试)

sample\_nnie\_main(神经网络硬件加速单元测试)

————————————————

版权声明：本文为CSDN博主「壹贰四六玖」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：<https://blog.csdn.net/qq_40421682/java/article/details/101603190>

编译与运行 sample

请参考海思Hi3519A开发（5.梳理海思文档与运行sample代码）

sample\_venc(视频数据编码)

# ./sample\_venc

Usage : ./sample\_venc [index]

index:

0) H.265e@4K@120fps + H264e@1080p@30fps.

1) H.265e@4K@60fps + H264e@4K@60fp.

2) Lowdelay:H.265e@4K@30fps.

3) IntraRefresh:H.265e@4K@60fps + H264e@4K@60fps.

4) Qpmap:H.265e@4K + H264e@4K@60fps.

5) Mjpeg@4K@60fps +Jpeg@4K.

1

2

3

4

5

6

7

8

9

shell 说明：

运行sample\_venc (录制) 视频编码样例

使用方法: ./sample\_venc [index]

参数<索引>

0) h265编码,4096×2160分辨率,每秒120帧 + h264编码,1920x1080分辨率,每秒30帧

1) h265编码,4096×2160分辨率,每秒60帧 + h264编码,4096×2160分辨率,每秒60帧

2) Lowdelay(低延时属性)：h265编码,4096×2160分辨率,每秒30帧

3) IntraRefresh(P 帧帧内刷新)：h265编码,4096×2160分辨率,每秒60帧 + h264编码,4096×2160分辨率,每秒60帧

4) Qpmap(是一种码率控制模式)：h265编码,4096×2160分辨率 + h264编码,4096×2160分辨率,每秒60帧

5) Mjpeg(Mjpeg协议编码方式)的4096×2160分辨率,每秒60帧 + Jpeg(Jpeg的编码)的4096×2160分辨率

sample\_vdec(视频数据解码)

# ./sample\_vdec

Usage : ./sample\_vdec <index> <IntfSync>

index:

0) VDEC(H265)-VPSS-VO

1) VDEC(H264)-VPSS-VO

2) VDEC(JPEG->YUV)-VPSS-VO

3) VDEC(JPEG->RGB)

IntfSync :

0) VO HDMI 4K@30fps.

1) VO HDMI 1080P@30fps.

1

2

3

4

5

6

7

8

9

10

shell 说明：

运行sample\_vdec (播放) 视频解码样例

使用提示： ./sample\_vdec <索引> <Vo接口时序类型>

参数<索引>:

0) VDEC解码器输入(H265的编码格式图片/视频)—>VPSS(视频输入缓存块)—>VO(图片/视频输出)

1) VDEC解码器输入(H264的编码格式图片/视频)—>VPSS(视频输入缓存块)—>VO(图片/视频输出)

2) VDEC解码器输入(JPEG数据)—>VPSS(视频输入缓存块)—>VO(YUV图片/视频输出)

3) VDEC解码器输入JPEG解码成RGB数据

参数<IntfSync>:

0) 输出到HDMI 4096×2160分辨率,每秒30帧

1) 输出到HDMI 1920x1080分辨率,每秒30帧

sample\_audio(音频相关)

# ./sample\_audio

/Usage:./sample\_audio <index>/

index and its function list below

0: start AI to AO loop

1: send audio frame to AENC channel from AI, save them

2: read audio stream from file, decode and send AO

3: start AI(VQE process), then send to AO

4: start AI to AO(Hdmi) loop

5: start AI to AO(Syschn) loop

6: start AI to Extern Resampler

channel:

0: mic0 input

1: mic1 input

1

2

3

4

5

6

7

8

9

10

11

12

13

shell 说明：

运行sample\_audio 音频 （编码/解码）样例

使用提示：./sample\_audio <索引>

以下内容是索引对应的功能

0) 循环音频从输入到输出 (话筒功能)

1) 通过音频输入发送音频帧到音频编码通道，保存起来 (录音功能)

2) 从文件读取音频流，解码然后发送到输出 (播放功能)

3) 通过音频输入声音质量增强处理，然后输出音频 (话筒功能)

4) 循环音频输入到HDMI通道音频输出 (话筒功能)

5) 循环音频输入到系统音通道音频输出 (话筒功能)

6) 通过音频输入重新采样 (录音功能)

音频输入通道

0) mic0输入(HeadPhone)

1) mic1输入(插针)

sample\_snap(拍照)

# ./sample\_snap

Usage : ./sample\_snap <index>

index:

0)double pipe offline, normal snap.

1

2

3

4

shell 说明：

运行sample\_snap 拍照

使用说明：./sample\_snap <索引>

参数<索引>

0)双 pipe 离线模式普通拍照

sample\_dpu\_main(Depth Process Unit,深度图)

# ./sample\_dpu\_main

Usage : ./sample\_dpu\_main <index>

index:

0) VI->VPSS->RECT->MATCH.

1) FILE->RECT->MATCH.

1

2

3

4

5

shell 说明：

运行 sample\_dpu\_main

使用方法：./sample\_dpu\_main <索引>

<索引>

0) DPU(Depth Process Unit)对输入的左图像和右图像经过校正和匹配计算得出深度图

1) DPU(Depth Process Unit)对读取的文件的左图像和右图像经过校正和匹配计算得出深度图

sample\_avs(Any View Stitching,全景拼接)

# ./sample\_avs

Usage : ./sample\_avs <index>

index:

0) 2 fisheye stitching, Normal projection.

1) 4 fisheye stitching, Cube map.

2) 4 pic no blend stitching.

3) 2 fisheye stitching, Image stabilizing.

4) Generate lut.

1

2

3

4

5

6

7

8

shell 说明：

运行 sample\_avs

使用说明： ./sample\_avs <index>

<index>

0) 2个CAM 鱼眼拼接正常显示模式

1) 4个CAM 鱼眼拼接呈现立体显示效果

2) 4个CAM 图片无混合拼接

3) 目前暂不支持

4)生成LUT表

sample\_fisheye(鱼眼模式)

# ./sample\_fisheye

Usage : ./sample\_fisheye <index> <vo intf> <venc type>

index:

0) fisheye 360 panorama 2 half with ceiling mount.

1) fisheye 360 panorama and 2 normal PTZ with desktop mount.

2) fisheye 180 panorama and 2 normal dynamic PTZ with wall mount.

3) fisheye source picture and 3 normal PTZ with wall mount.

4) nine\_lattice preview(Only images larger than or equal to 8M are supported).

vo intf:

0) vo HDMI output, default.

1) vo BT1120 output.

venc type:

0) H265, default.

1) H264.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

shell 说明：

运行 ./sample\_fisheye

使用说明 ./sample\_fisheye <index> <vo intf> <venc type>

index:

0) 2个“半顶装模式”的鱼眼360°全景视频

1) 2个普通“地装的PTZ”鱼眼360°全景视频

2) 2个普通“壁装的PTZ”鱼眼180°全景视频

3) 三个鱼眼原画的普通PTZ 壁装视频

4) 九格预览视频(仅支持大于或等于8m的图像)

vo intf:

0) 默认HDMI输出

1) BT1120输出

venc type:

0) 默认H265编码

1) H264编码

sample\_vio(视频输入输出)

# ./sample\_vio\_main

Usage : ./sample\_vio\_main <index> <intf>

index:

0)VI(Online) - VPSS(Online) - VO.

1)VI(Offline)- VPSS(Offline) - VO. LDC+DIS+SPREAD.

2)VI(Offline)- VPSS(Online) - VO. Double pipe.

3)VI(Online)- VPSS(Offline) - VO. Double chn.

4)Resolute Ratio Switch.

5)GDC - VPSS LowDelay.

6)Double WDR Pipe.

7)FPN Calibrate & Correction.

8)WDR Switch.

9)90/180/270 Rotate.

10)Mipi Demux Yuv.

11)UserPic.

intf:

0) vo HDMI output, default.

1) vo BT1120 output.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

shell 说明：

运行 sample\_vio

使用方法：./sample\_vio\_main <index> <intf>

index:

0) 在线输入–> 在线视频处理 -->输出

1) 离线输入–> 离线视频处理 -->输出 镜头畸变校正视频防抖和展宽

2) 离线输入–> 在线视频处理 -->输出 双管道

3) 在线输入–> 离线视频处理 -->输出 双通道

4) Resolute Ratio Switch

5) 几何畸变矫正 -->视频处理 低延时

6) 双WDR管道

7) FPN标定&矫正

8) WDR 开关

9) 90/180/270 旋转

10) Mipi Demux Yuv

11) 设置用户图片

intf:

0) 默认HDMI输出

1) BT1120输出

sample\_vgs(Video Graphics Sub-System,视频图形子系统)

# ./sample\_vgs

Usage: ./sample\_vgs <index>

index:

0) FILE -> VGS(Scale) -> FILE.

1) FILE -> VGS(Cover+OSD) -> FILE.

2) FILE -> VGS(DrawLine) -> FILE.

3) FILE -> VGS(Rotate) -> FILE.

1

2

3

4

5

6

7

shell 说明：

运行 sample\_vgs

使用说明：./sample\_vgs <index> <intf>

index:

0) 文件–> 视频图像子系统(缩放) -->文件

1) 文件–> 视频图像子系统(Cover+OSD) -->文件

2) 文件–> 视频图像子系统(画线) -->文件

3) 文件–> 视频图像子系统(旋转) -->文件

sample\_tde(Two Dimensional Engine,利用硬件为OSD和GUI提供快速的图形绘制功能)

# ./sample\_tde

Usage : ./sample\_tde <intf>

intf:

0) vo BT1120 output, default.

1) vo HDMI output.

1

2

3

4

5

shell 说明：

运行 sample\_tde

使用说明：./sample\_tde <intf>

intf:

0) 默认BT1120输出演示

1) HDMI输出演示

sample\_hifb(Hisilicon Framebuffer,基于Linux FB 基本功能扩展了一些图形层控制功能)

# ./sample\_hifb

Usage : ./sample\_hifb <index> <device> <intf>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*index\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

please choose the case which you want to run:

0: ARGB8888 standard mode

1: ARGB1555 BUF\_DOUBLE mode

2: ARGB1555 BUF\_ONE mode

3: ARGB1555 BUF\_NONE mode

4: ARGB1555 BUF\_ONE mode with compress

5: ARGB8888 BUF\_ONE mode with compress

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*device\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0) VO device 0#, default.

1) VO device 1#.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*intf\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0) VO HDMI output, default.

1) VO BT1120 output.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

shell 说明：

运行 sample\_hifb

使用说明：./sample\_hifb <index> <device> <intf>

<index>

请选择一下您想运行的情况之一

0: ARGB8888像素格式标准模式

1: ARGB1555像素格式双缓存模式

2: ARGB1555像素格式单缓存模式

3: ARGB1555像素格式无缓存模式

4: ARGB1555像素格式带压缩的单缓存模式

5: ARGB8888像素格式带压缩的单缓存模式

<device>

0) 默认输出到超清显示设备DHD0

1) 输出到高清显示设备DHD1(暂未支持)

<intf>

0) 默认HDMI输出

1) BT1120输出(暂未支持)

sample\_awb\_calibration(自动白光平衡测量)

# ./sample\_awb\_calibration

Usage : ./sample\_awb\_calibration <mode> <intf1> <intf2> <intf3>

mode:

0) Calculate Sample gain.

1) Adjust Sample gain according to Golden Sample.

intf1:

The value of Rgain of Golden Sample.　深红色区域的锐化增益控制。

intf2:

The value of Bgain of Golden Sample.

intf3:

The value of Alpha ranging from 0 to 1024 (The strength of adusting Sampe Gain will increase with the value of Alpha) .

1

2

3

4

5

6

7

8

9

10

11

shell 说明：

运行 sample\_awb\_calibration

使用方法：./sample\_awb\_calibration <mode> <intf1> <intf2> <intf3>

<mode>

0) 计算例子中的增益

1) 根据标准例子来校正例子中的增益

intf1:

标准例子中的深红色区域的锐化增益控制值

intf2:

标准例子中的深蓝色区域的锐化增益控制值

intf3:

0~1024范围的初始值(采样数据的增益强度会随着初始值的值增加而增加)

sample\_ive\_main(智能加速引擎)

~ # ./sample\_ive\_main

Usage : ./sample\_ive\_main <index> [complete] [encode] [vo]

index:

0)BgModel,<encode>:0, not encode;1,encode.<vo>:0,not call vo;1,call vo.(VI->VPSS->IVE->VGS->[VENC\_H264]->[VO\_HDMI]).

1)Gmm,<encode>:0, not encode;1,encode.<vo>:0,not call vo;1,call vo.(VI->VPSS->IVE->VGS->[VENC\_H264]->[VO\_HDMI]).

2)Occlusion detected.(VI->VPSS->IVE->VO\_HDMI).

3)Motion detected.(VI->VPSS->IVE->VGS->VO\_HDMI).

4)Canny,<complete>:0, part canny;1,complete canny.(FILE->IVE->FILE).

5)Gmm2.(FILE->IVE->FILE).

6)MemoryTest.(FILE->IVE->FILE).

7)Sobel.(FILE->IVE->FILE).

8)Ann.(FILE->IVE->STDIO).

9)St Lk.(FILE->IVE->FILE).

a)Svm.(FILE->IVE->STDIO).

b)Cnn.(FILE->IVE->STDIO).

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

shell 说明：

运行 sample\_ive\_main

使用说明: ./sample\_ive\_main <index> [complete] [encode] [vo]

index:

0) 背景模型

1) 高斯模型Gmm

2) 遮挡检测

3) 运动检测

4) 边缘检测

5) 高斯模型Gmm2

6) 内存测试

7) Sobel算子分割

8) 图像检索Ann

9) St和LK光流法

a) 分类器SVM

b) 神经网络Cnn

sample\_dis(Digital Image Stabilization,数字稳像)

# ./sample\_dis

Usage : ./sample\_dis <index> <intf>

index:

0)DIS-4DOF\_GME.VI-VO VENC.

1)DIS-6DOF\_GME.VI-VO VENC.

intf:

0) vo HDMI output, default.

1) vo BT1120 output.

1

2

3

4

5

6

7

8

shell 说明：

运行 sample\_dis

使用说明: ./sample\_dis <index> <intf>

index:

0) DIS-4DOF\_GME(四自由度 GME 算法,不使用陀螺仪)，输入－输出　同时H256格式录像(存储在当前)

1) DIS-6DOF\_GME(六自由度 GME 算法,不使用陀螺仪)，输入－输出　同时H256格式录像(存储在当前)

intf:

0) HDMI 输出

1) BT1120 输出

sample\_dsp\_main(DSP测试)

~ # ./sample\_dsp\_main

1

shell 说明：

四个DSP之DSP0的出图测试。

sample\_nnie\_main(神经网络硬件加速单元测试)

神经网络，特别是深度学习卷积神经网络进行加速处理的硬件单元测试。

# ./sample\_nnie\_main

Usage : ./sample\_nnie\_main <index>

index:

0) RFCN(VI->VPSS->NNIE->VGS->VO).

1) Segnet(Read File).

2) FasterRcnnAlexnet(Read File).

3) FasterRcnnDoubleRoiPooling(Read File).

4) Cnn(Read File).

5) SSD(Read File).

6) Yolov1(Read File).

7) Yolov2(Read File).

8) LSTM(Read File).

9）Pvanet(Read File).

a) Rfcn(Read File).

1

2

3

4

5

6

7

8

9

10

11

12

13

14

shell 说明：

运行 sample\_nnie\_main

使用说明: ./sample\_nnie\_main <index>

index:

0) RFCN模型

1) 可训练的图像分割Segnet

2) 深度学习的目标检测Alexnet

3) 深度学习的目标检测DoubleRoiPooling

4) 神经网络Cnn

5) 可训练的SSD模型处理

6) 神经网格模型Yolov1

7) 神经网格模型Yolov2

8) LSTM模型

9) Pvanet网络

a) Rfcn目标检测

————————————————

版权声明：本文为CSDN博主「壹贰四六玖」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/qq\_40421682/java/article/details/101603190