



DIS 调试指南

文档版本 00B09
发布日期 2018-11-13

杭州雄迈信息技术有限公司Hi3519A V100R001C02SPC010杭州雄迈信息技术有限公司Hi3519A V100R001C02SPC010杭州雄迈信息

版权所有 © 深圳市海思半导体有限公司 2018。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：深圳市龙岗区坂田华为基地华为电气生产中心 邮编：518129

网址：<http://www.hisilicon.com>

客户服务电话：+86-755-28788858

客户服务传真：+86-755-28357515

客户服务邮箱：support@hisilicon.com



前言

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3559A	V100ES
Hi3559A	V100
Hi3559C	V100
Hi3519A	V100
Hi3556A	V100
Hi3516C	V500
Hi3516D	V300
Hi3559	V200
Hi3556	V200



说明

未有特殊说明，Hi3559CV100 与 Hi3559AV100 内容一致。

未有特殊说明，Hi3556AV100 与 Hi3519AV100 内容一致。

未有特殊说明，Hi3516DV300 与 Hi3516CV500 内容一致。

未有特殊说明，Hi3559V200 与 Hi3516CV500 内容一致。

未有特殊说明，Hi3556V200 与 Hi3516CV500 内容一致。

读者对象




本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师



符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 危险	表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害。
 警告	表示有中度或低度潜在危险，如果不能避免，可能导致人员轻微或中等伤害。
 注意	表示有潜在风险，如果忽视这些文本，可能导致设备损坏、数据丢失、设备性能降低或不可预知的结果。
 窍门	表示能帮助您解决某个问题或节省您的时间。
 说明	表示是正文的附加信息，是对正文的强调和补充。

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 00B09 (2018-11-13)

第 9 次临时版本发布。

2.2 小节的 u32RollingShutterCoef 涉及修改

文档版本 00B08 (2018-09-04)

第 8 次临时版本发布。

添加 Hi3516CV500 和 Hi3516DV300 相关内容

文档版本 00B07 (2018-07-30)

第 7 次临时版本发布。

2.3.1 小节涉及修改。

文档版本 00B06 (2018-01-30)

第 6 次临时版本发布。

1.3 小节涉及修改



文档版本 00B05 (2017-11-15)

第 5 次临时版本发布。

添加 Hi3559AV100 相关内容

文档版本 00B04 (2017-09-20)

第 4 次临时版本发布。

1.2 小节，GME 算法描述涉及修改。

1.3 小节，图 1-2 涉及修改。

2.1 小节，u32CropRatio 增加注意。

文档版本 00B03 (2017-06-30)

第 3 次临时版本发布。

2.2 小节，u32HorizontalLimit 和 u32VerticalLimit 涉及修改

2.3.5 小节涉及修改

文档版本 00B02 (2017-05-27)

第 2 次临时版本发布。

2.1 小节，新增图 2-1

2.2.6、2.2.12 小节涉及修改

新增 2.2.7、2.2.8、2.2.11、2.3 小节

文档版本 00B01 (2017-04-28)

第 1 次临时版本发布。



目 录

前 言.....	i
1 概述.....	1
1.1 概念.....	1
1.2 DIS 基本原理	1
1.3 DIS 的实现	2
2 DIS 开发应用	4
2.1 DIS 使用	4
2.2 参数设置.....	4
2.3 陀螺仪使用	8
2.3.1 陀螺仪算法的流程.....	8
2.3.2 驱动和 Sample 代码.....	8
2.3.3 适配陀螺仪和图像坐标系方向.....	9
2.3.4 确定 u32ViewAngle 和 u32Timelag	12
2.3.5 初始化和启动陀螺仪.....	12
2.3.6 陀螺仪的配置.....	13
2.3.7 陀螺仪的 proc 信息	13



插图目录

图 1-1 DIS 原理图 2

图 1-2 VI 通道功能框图 2

图 2-1 DIS 使用流程 4

图 2-2 视场角示意图 7

图 2-3 DIS 获取陀螺仪数据原理图 9

图 2-4 图像坐标系 10

图 2-5 Bosch BM160 陀螺仪坐标系 10

图 2-6 陀螺仪安装位置 1 11

图 2-7 陀螺仪安装位置 2 12

图 2-8 Buffer 数据图 12



表格目录

表 2-1 参数说明 1 13

表 2-2 参数说明 2 13



1 概述

摄像机在拍摄视频时由于环境或人为的影响，视频会出现抖动或不稳定现象，影响视频观看，比如在监控场景中由于摄像机可能受到风吹或汽车经过引起的振动，手持运动 DV 受到人为的影响，行车记录仪受到汽车的抖动，都会引起视频抖动，为了让客户观看更加稳定和更加舒服的视频，需要尽可能消除视频的抖动。

1.1 概念

DIS (Digital Image Stabilization) 数字图像防抖

DIS 是对图像进行数字处理的过程，采用防抖算法计算出当前图像的运动偏移，然后根据计算得到的运动偏移对当前图像进行平移、旋转等变换，从而起到防抖的效果。

1.2 DIS 基本原理

DIS 的基本原理是根据运动偏移对图像的二维仿射变换过程。仿射变换包括对图像的平移、旋转、放大和错切（可通俗理解为平行四边形变换）等，此变换可以用 3x3 的矩阵来表示。

$$\begin{bmatrix} \mu \\ v \\ \omega \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

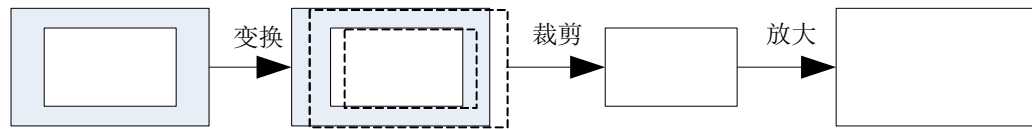
$$x' = \frac{\mu}{\omega}; \quad y' = \frac{v}{\omega}$$

3x3 矩阵就是 DIS 算法中要计算的运动偏移，(x,y)表示原来图像的坐标位置，(x',y')是变换后的图像的坐标位置。

在对图像进行图像变换操作时，会改变图像像素的位置，使得图像的边缘可能会超出原有图像的宽高，会被平移出整个画面的位置。因此在 DIS 防抖时要对图像进行裁剪和放大。DIS 防抖是要在完成变换操作后对图像按照一定的裁剪比例裁剪掉图像的边缘，然后再按原有宽高放大裁剪之后的图像，如图 1-1 所示。



图1-1 DIS 原理图



DIS 计算运动偏移有三种算法：

- GME 算法

GME(Global Motion Estimation)算法是通过提取图像特征，计算当前帧图像和参考帧图像之间的运动偏移。采用 GME 算法处理后的图像较稳定，有较好的防抖效果，但当画面中大面积拍摄物体在移动时，画面也会出现背景拖拽现象。这是因为 GME 无法完全区分出画面物体移动还是摄像机移动，从而可能造成误判。另外在低照度情况下，由于图像的特征模糊，GME 算法防抖效果存在下降的可能。

- 陀螺仪算法

陀螺仪算法是根据陀螺仪产生的数据计算当前帧图像的运动偏移，采用陀螺仪算法能够较好解决误判和低照度情况下无防抖效果等现象。

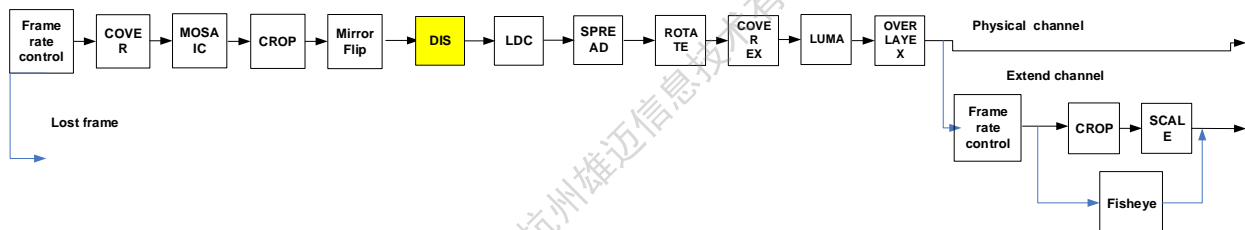
- 混合算法（GME+陀螺仪）

由于陀螺仪算法本身的局限性，使用陀螺仪算法的防抖效果较弱。于是就产生了混合算法。混合算法是同时采用 GME 和陀螺仪算法计算运动偏移。

1.3 DIS 的实现

DIS 功能集成在 VI 模块，如图 1-2 所示。

图1-2 VI 通道功能框图





注意

- DIS 只支持在物理通道上运行。
- **DIS** 视频输入图像格式支持线性 Semi-planar420 和单分量，只支持非压缩图像。开启 DIS 后，不能在 VI 中进行 flip 和 mirror 操作。
- DIS 视频输入图像幅型比（宽高比）支持范围为 16:3~16:27
- DIS 处理过程中需要使用 VGS/GDC 模块，如果多个模块调用 VGS 或 GDC，可能因 VGS、GDC 性能不足而导致 DIS 出现丢帧的现象。

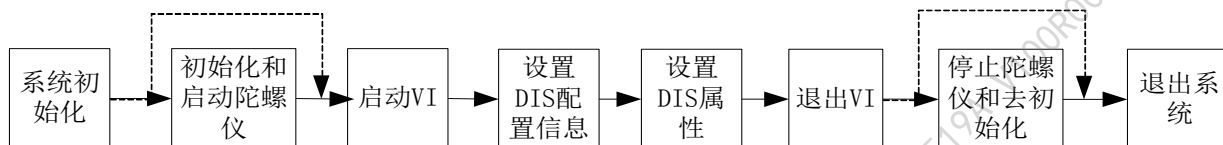


2 DIS 开发应用

2.1 DIS 使用

DIS 相关 API 和参数说明请参见《HiMPP V4.0 媒体处理软件开发参考》的视频输入章节。具体实现可参考 DIS 的 sample，DIS 使用流程如图 2-1 所示。

图2-1 DIS 使用流程



2.2 参数设置

在启动 DIS 前需要先设置 DIS 配置信息和属性等相关参数，参数的不同值会不同程度的影响防抖效果。这里主要介绍几个重要影响防抖效果的参数设置。

u32CropRatio

DIS 输出图像的裁剪比例。其取值范围为[50, 98]。通常设置为 80，即防抖处理后只输出图像的 80%。假设输入图像宽高为 1920*1080，设置 u32CropRatio 为 80，即裁掉输入图像左右边缘和上下边缘的各 10%，裁剪后的图像宽为 $(1920 - 2 * 1920 * 10\%) = 1536$ ，高为 $(1080 - 2 * 1080 * 10\%) = 864$ 。**注意：如果裁剪后宽高不为偶数，则向下去偶。**

- 当视频输入分辨率大于或等于 1920*1080 时，最小支持 u32CropRatio 为 50。
- 当输入分辨率小于 1920*1080 时，最小支持 u32CropRatio 为 70。

enMode

在 DIS 算法中会使用到自由度（dof, degree of freedom）的概念。



- 从用户角度来看：
自由度的概念就是三维空间，X、Y、Z 三个轴，每个轴可以有两种动作：平动、转动。一共产生 6 种运动。这也是通常所说的 6 轴防抖。
- 从算法角度来看：
自由度表示仿射变换的 3x3 矩阵中使用的算子数目。

不同自由度数目对图像进行仿射变换的操作也不一样。

4_DOF 与 6_DOF 的区别：

- 4_DOF：算法中使用了 4 个算子，主要是对图像进行平移、旋转和放大操作。相对于 6_DOF，少了 2 个算子，计算算子越少，也更不容易造成误判，其也能较好的防住大面积物体移动造成的背景拖拽现象，rollingshutter 现象较明显。
- 6_DOF：算法中使用了 6 个算子，主要是对图像进行平移、旋转、放大、改变图像宽高比及错切。优点是防抖效果较好，能够对平行四边形进行校正，缺点是容易引起背景拖拽等异常现象。

enMotionLevel

Camera 的运动级别，分为：Low、Normal 和 High。

- Low 是指低级别运动，镜头小幅度运动。
- Normal 是指正常级别运动，镜头正常幅度运动。
- High 是指高级别运动，镜头大幅度运动。

通常设置为 Normal，请根据实际运动幅度进行调整。



说明

Hi3516CV500 不支持 Low 模式。

enPdtType

防抖支持的产品形态。当前支持三种产品形态，分别是 IPC、DV 和无人机。请根据实际产品形态配置产品类型。

bCameraSteady

镜头是否是固定静止的开关。该参数只在 IPC 产品形态才会起作用，在 DV 和无人机产品形态下，该参数不起作用，默认设置为 HI_FALSE。

as32RotationMatrix

旋转矩阵，3x3 的矩阵，MATRIX_NUM 为 9。用于 Gyro sensor 坐标系和图像坐标系的方向的转换，算法参考的是图像的坐标系，陀螺仪的不同安装位置对应坐标系方向不一样，因此需要将陀螺仪坐标系与图像坐标系方向进行转换。另外在安装陀螺仪时请确保陀螺仪芯片与 image sensor 位置保持水平或垂直方向。

陀螺仪坐标系方向与图像坐标系方向通过旋转矩阵进行转换。假设陀螺仪数据为 (Xg, Yg, Zg)，算法使用的陀螺仪数据为 (Xa, Ya, Za)。



$$\begin{bmatrix} Xa \\ Ya \\ Za \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & l \end{bmatrix} \begin{bmatrix} Xg \\ Yg \\ Zg \end{bmatrix}$$

旋转矩阵 `as32RotationMatrix[MATRIX_NUM]` 的 9 个参数分别对应[a,b,c,d,e,f,g,h,l]。计算 9 个参数的具体方法请参见 [2.3.3 适配陀螺仪和图像坐标系方向](#)。

u32GyroOutputRange

`u32GyroOutputRange` 为陀螺仪数据的角度范围，其取值范围为[0, 360]。我们当前陀螺仪使用的量程是 250。

u32GyroDataBitWidth

`u32GyroDataBitWidth` 为陀螺仪数据的位宽，其取值范围为[0, 32]，当前陀螺仪数据的位宽是 15bit，即陀螺仪输出数据的值为[-32768, 32768]。

u32MovingSubjectLevel

用于判断拍摄物体是否是运动的级别，取值范围为[0,6]，该参数主要是防止背景拖拽。背景拖拽和防抖效果两者是相互权衡的。

- 值越小时，运动过程中越稳定，但更容易出现背景拖拽情况；
- 值较大时，运动过程中防抖效果较弱，但是能够较好的改善背景拖拽现象；

u32RollingShutterCoef

校正 `rollingshutter` 强度的参数，取值范围为[0,1000]。此参数适用于相机一直朝向一个方向长时运动的场景，如火车上拍摄外景等。对于来回抖动带来的 `rolling shutter` 现象，算法会自适应的检测和做相关的矫正并改善 `rolling shutter` 现象，建议配置此参数为 0。

u32ViewAngle

在使用到陀螺仪时，正确设置视场角的大小很重要，如果设置视场角与实际视场角相差太大，有可能会出现加剧抖动等异常情况。视场角大小的取值范围是[100,1380]，视场角的输入值为整数，其表示镜头的视场角大小为 `u32Vangle/10`。例如：要表示 80.7 度的值，就要将 `u32Vangle` 赋值为 807。

目前算法中使用的是水平视场角，下面介绍水平视场角的计算，如[图 2-2](#) 所示。

视场角的计算公式如下：

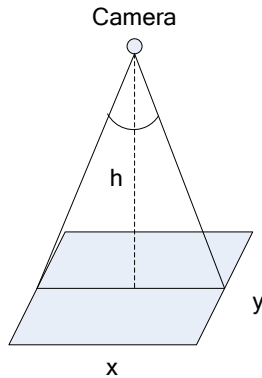
$$\text{ViewAngle} = 2 * \{ \arctan(x / (2 * h)) * (180 / \pi) \}$$

- `x` 表示镜头能看到视场范围的宽的实际距离；
- `y` 表示镜头能看到视场范围的高的实际距离；
- `h` 是镜头距离成像画面的实际距离。



假设 x 为 55cm, y 为 73cm, h 为 70cm, 计算 \arctan 时用弧度计算, 根据上述公式计算得到的水平视场角为 42.9 度, 因此配置 `u32ViewAngle` 为 429。

图2-2 视场角示意图



u32Timelag

帧起始时间和陀螺仪数据开始采集时间的时间差。取值范围为[0, 2000000], 单位是 us。u32Timelag 在使用陀螺仪时也至关重要。u32Timelag 仅在 gyro/hyb 模式有效, 一般配置为一帧时间, 要达到比较好的效果, 需要进行微调。如果 u32Timelag 设置过大或者过小, 有可能出现加剧抖动等异常情况。典型值如下:

- 帧率 30 帧时 u32Timelag 为 33333us;
- 帧率为 60 时 u32Timelag 为 16666us;
- 帧率为 120 时 u32Timelag 为 8333us;

其他帧率情况下适当进行调整, 以达到最好的防抖效果。

u32HorizontalLimit 和 u32VerticalLimit

水平偏移和垂直偏移限制, 取值范围[0,1000]。当大面积物体经过引起背景拖拽的水平偏移超过一定幅度时就不进行防抖。偏移幅度计算: $2047 * u32HorizontalLimit / 1000$ 。

该参数需和 bCameraSteady 配合使用, 该参数只在 bCameraSteady 为 HI_TRUE 时生效。当 bCameraSteady 为 HI_FALSE 时, 默认设置为 1000。

bStillCrop

该开关的作用是关闭 DIS 防抖效果, 但图像依旧保持裁剪比例输出。打开该开关后, DIS 输出图像没有防抖效果, 但是输出图像的裁剪比例还是跟有防抖效果的输出图像的裁剪比例一致的。通常该参数设置为 HI_FALSE, 如有需要时设置该值为 HI_TRUE。



2.3 陀螺仪使用

在防抖中使用陀螺仪的目的主要是：

- 防止背景拖拽问题

在很多情况下 GME 算法是无法判断是前景在动还是镜头在动。例如当有大面积物体在镜头前移动，而镜头是静止的。此时算法可能会发生误判，将前景运动判断为镜头运动会进行防抖，从而造成背景拖拽现象。陀螺仪可以反映机器自身的运动状态，增加陀螺仪可以很好的弥补了该缺陷。

- 在低照度或者特征点较少的场景中有防抖效果

在低照度情况下由于图像背景较暗，对于 GME 算法来说无法提取到特征点，从而低照度情况下几乎无防抖效果，采用陀螺仪的话上述问题就迎刃而解。

2.3.1 陀螺仪算法的流程

使用陀螺仪算法流程如下：

步骤 1. 使用陀螺仪相关算法前，须确保单板上带有陀螺仪芯片且可用。

步骤 2. 在启动 DIS 功能前，需要先加载 motionsensor_chip/motionsensor_mng/hi3559av100_motionfusion 三个驱动，并且保证 MotionSensor 已经开始工作并产生数据。

注意：

- 加载陀螺仪驱动，在加载 motionsensor_chip 驱动前需要先加载 motionsensor_mng 驱动。每次加载 motionsensor_mng 驱动后都需要重新加载 motionsensor_chip 驱动；
- MotionSensor 的启动流程为：初始化-设置 MotionSensor 的参数-启动 MotionSensor；MotionSensor 的停止流程为：停止 MotionSensor 工作-去初始化。
- 建议在使能 DIS 同时模式为 DIS_MODE_GYRO 之前启动 MotionSensor 工作；在停止 VI 工作之后停止 MotionSensor 工作。

步骤 3. 适配陀螺仪坐标系和图像坐标系的方向，配置正确的旋转矩阵 as32RotationMatrix[MATRIX_NUM]。

步骤 4. 确定正确的镜头视场角 u32ViewAngle 和 u32Timelag。

步骤 5. 启动 DIS 前，需先初始化和启动陀螺仪。

步骤 6. 关闭 DIS 后，再停止陀螺仪，最后再系统退出。

----结束

具体实现请参考 DIS 的 sample 中使用陀螺仪相关部分。

2.3.2 驱动和 Sample 代码

SDK 发布包里有 bosch 陀螺仪的驱动代码，其他型号的陀螺仪请参考上述驱动代码自行对接。

代码路径:

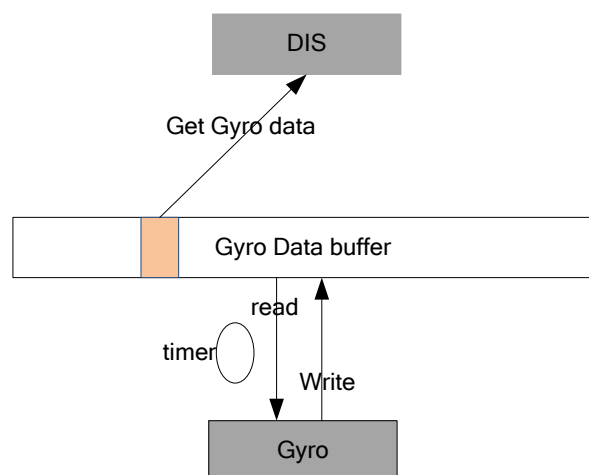
Bosch driver: \drv\extdrv\mpu_bosch\

Sample: \mpp\sample\dis\

使用时只需要在 extdrv 目录下执行 make 命令即可在 \mpp\ko\ 目录下获得 ko 文件。默认 \mpp\ko\load 脚本是不加载陀螺仪驱动, 请结合实际情况修改。

DIS 获取陀螺仪数据的原理如图 2-3 所示。

图2-3 DIS 获取陀螺仪数据原理图



陀螺仪数据存在分配的 Gyro Data buffer 中。启动陀螺仪驱动后, 陀螺仪驱动内部会启动个定时器, 不断从陀螺仪 fifo 中读取陀螺仪数据并为每组数据打上时间戳, 然后将数据写到 Gyro Data buffer。DIS 驱动根据每帧的起始时间戳和结束时间戳从 Gyro Data buffer 中获取对应时间段内的陀螺仪数据用于防抖处理。

2.3.3 适配陀螺仪和图像坐标系方向

安装陀螺仪芯片时, 要确保陀螺仪芯片是正装, 即与 image sensor 保持平行或者垂直。

使用陀螺仪算法时, 镜头的移动信息靠陀螺仪提供, 因此陀螺仪数据的准确性是至关重要的。陀螺仪不同的安装位置, 其对应的坐标系方向也不同。

使用 DIS 陀螺仪相关算法时, 首先要正确匹配陀螺仪和图像的坐标方向。

图 2-4 为图像坐标系, 为了便于描述清楚, 图中使用手机屏幕进行说明。

- z 轴与图像画面垂直, z 轴指向人眼的方向是正方向;
- x 轴和 y 轴分别是水平和垂直方向, 对应着图像的宽高。

DIS 算法参考的是图像的坐标系。Bosch BM160 陀螺仪坐标系如图 2-5 所示。

图2-4 图像坐标系

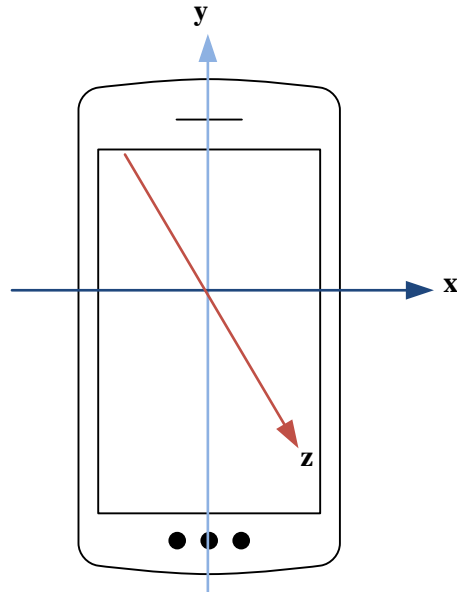
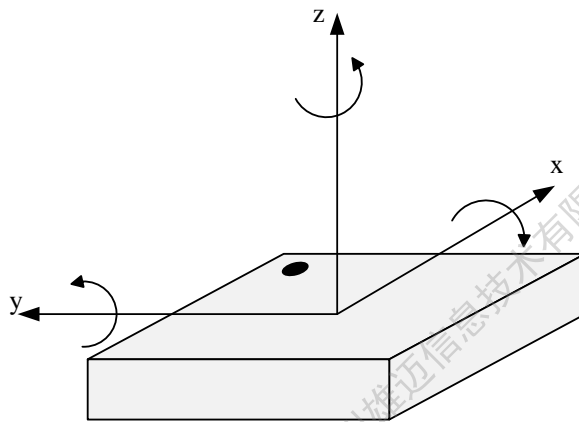


图2-5 Bosch BM160 陀螺仪坐标系



下面从陀螺仪的 2 种不同安装位置方式来说明下坐标系方向如何转换，其他安装位置，请以此类推。黑点表示陀螺仪芯片管脚 1 的位置。

- 陀螺仪安装位置 1

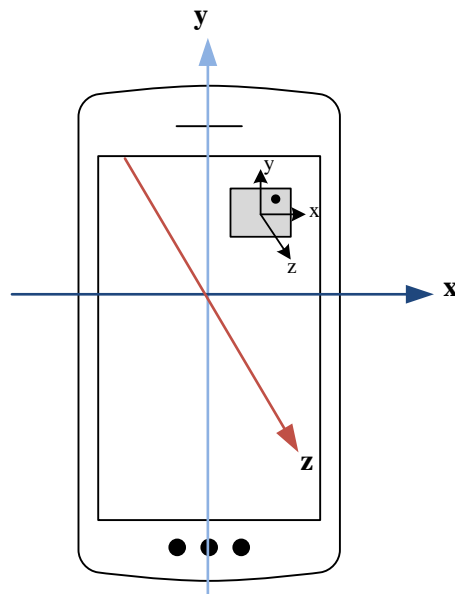
当陀螺仪的安装位置如图 2-6 所示时，此时陀螺仪的坐标系和图像坐标系方向是一致的。陀螺仪获取的数据为 (X_g, Y_g, Z_g) ，算法使用的陀螺仪数据为 (X_a, Y_a, Z_a) ，此时

$$\begin{bmatrix} Xa \\ Ya \\ Za \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Xg \\ Yg \\ Zg \end{bmatrix} = \begin{bmatrix} Xg \\ Yg \\ Zg \end{bmatrix}$$

所以设置 DIS Config 中的旋转矩阵为单位矩阵，即 $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ ，

as32RotationMatrix[MATRIX_NUM]的 9 个参数分别对应[1,0,0,0,1,0,0,0,1]。

图2-6 陀螺仪安装位置 1



• 陀螺仪安装位置 2

当陀螺仪的安装位置如图 2-7 所示时，此时陀螺仪的坐标系和图像坐标系方向不一致，需要转换下。陀螺仪获取的数据为(Xg,Yg,Zg)，算法使用的陀螺仪数据为(Xa,Ya,Za)，此时转换关系为：

$$\begin{bmatrix} Xa \\ Ya \\ Za \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Xg \\ Yg \\ Zg \end{bmatrix} = \begin{bmatrix} Yg \\ -Xg \\ Zg \end{bmatrix}$$

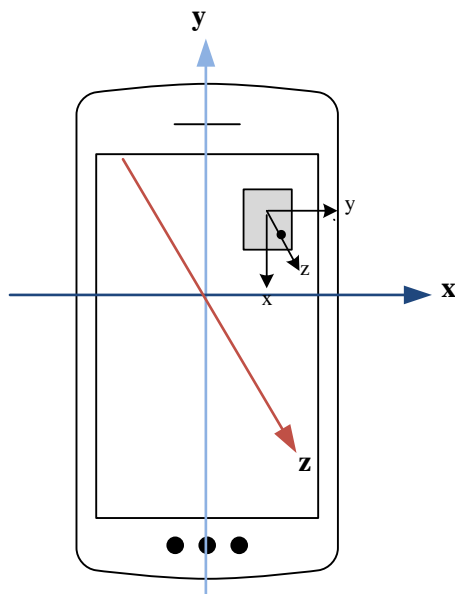
所以设置 DIS Config 中的旋转矩阵为：



$$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

as32RotationMatrix[MATRIX_NUM]的 9 个参数分别对应为[0,1,0,-1,0,0,0,0,1]。

图2-7 陀螺仪安装位置 2



2.3.4 确定 u32ViewAngle 和 u32Timelag

请参见 2.2 “参数设置” 中的 u32ViewAngle 和 u32Timelag 的参数计算方法。

2.3.5 初始化和启动陀螺仪

初始化陀螺仪主要是为陀螺仪数据分配 MMZ 内存用于保存陀螺仪数据。

陀螺仪输出的数据放在一个轮转 buffer 里面，算法根据帧中断来读取 buffer 中的陀螺仪数据。XYZ 轴数据与时间戳是一一对应的。

图2-8 Buffer 数据图

x	y	z	timestamp
---	---	---	-----------

MMZ 分配出来的 buffer 用于存储 4 部分数据：x、y、z 轴陀螺仪数据和时间戳。时间戳的数据类型长度是 8 个字节，XYZ 轴数据类型长度为 4 字节。每次帧中断来获取数据时都根据起始时间戳和结束时间戳在 buffer 段内查找数据。查找到符合条件的陀螺仪数据然后传给 DIS 算法使用。具体 buffer 的分配和大小请参考 sample。



2.3.6 陀螺仪的配置

陀螺仪的量程是 ± 250 的量程， ± 250 精度会更高一些，其防抖效果要比其他量程的防抖效果要好。

陀螺仪的数据采样频率（ODR）选择的是 800，陀螺仪输出数据的位宽是 15bit，陀螺仪数据范围是 $[-32768, 32768]$ 。

2.3.7 陀螺仪的 proc 信息

陀螺仪驱动的 proc 信息位置在 `/proc/mpu_info`。

陀螺仪 proc 信息如下：

```
~ # cat /proc/mpu_info
[mpu] Version:[MPU debug 0.0.0.1], Build Time[May 18 2017, 20:45:50]
-----MPUMODE-----
                gyro_fifo
-----GyroSensorName-----ODR-----RANGE-----BANDWIDTH-----
                BMI160           800           250           NORMAL
----buf_addr----buf_size----overflow----data_unmatch----overflowID----
data_unmatchID----
2064384          20000           0           0           0
0
```

参数说明如表 2-1 到表 2-2。

表2-1 参数说明 1

MPUMODE	陀螺仪模式
GyroSensorName	陀螺仪型号
ODR	数据采样频率
RANGE	陀螺仪量程
BANDWIDTH	带宽样式

表2-2 参数说明 2

buf_addr	轮转 buffer 的起始地址
buf_size	轮转 buffer 的总大小
block_size	Buffer 被均分 5 块后，块大小
overflow	Buffer 数据溢出次数
data_unmatch	帧中断来取陀螺仪数据没有找到匹配数据的次数



overflowID	Buffer 数据溢出的用户 ID
data_unmatchID	帧中断来取陀螺仪数据没有找到匹配数据的的用户 ID