

MicroDistPro 对位算法

Developer's Guide

修订记录

Rev	Data	Author	Description
1.0	20140530	Shuangyi	创建
1.1	20140610	Shuangyi	增加标定信息函数 SY_MD_Camera_Info 增加 4 相机对位函数 SY_MD_4Camera_Align
1.2	20170627	Shuangyi	删除部分旧内容
2.0	20171204	Shuangyi	增加图示说明, 丰富文档
2.1	20180710	Shuangyi	增加直线对位、特殊对位内容
2.2	20200709	Shuangyi	匹配库文件 20200609 版本

MicroDistPro 对位系统编程手册

修订记录.....	2
第 1 章 开发说明.....	5
1.1 函数库使用.....	5
1.2 应用指南.....	7
1.2.1 对位方式.....	7
1.2.2 标定和对位流程.....	8
1.2.3 对装有棱镜装置的视觉系统.....	8
1.2.4 程序流程图.....	9
第 2 章 函数说明.....	10
2.1 MicroDistPro 版本信息.....	11
2.2 对角补偿分量计算函数.....	11
2.3 物理量转换像素量函数.....	12
2.4 自标定函数.....	12
2.5 基于坐标系进行补偿的对位函数.....	13
2.6 基于每个相机进行补偿的对位函数.....	13
2.7 标定信息函数.....	14
2.8 相机映射学习相关函数.....	14
2.8.1 相机映射学习函数.....	14
2.8.2 相机映射函数 A.....	14
2.8.3 相机映射函数 B.....	15
2.9 测距函数.....	15
2.9.1 图像通道测距函数.....	15
2.9.2 图像通道测距函数 2.....	16
2.10 物理量转换为脉冲量.....	16
2.11 3 相机对位函数.....	16
2.12 4 相机对位函数.....	17
2.13 3 相机曝光机对位函数.....	18
2.14 4 相机曝光机对位函数.....	19
2.15 对角对位函数.....	19
2.15.1 对角对位函数 1.....	19
2.15.1 对角对位函数 2.....	20
2.15.3 对角对位函数 3.....	20
2.16 单点对位函数.....	21
2.17 4 相机直线对位.....	21
2.17.1 4 相机直线标定函数 1.....	21
2.17.2 4 相机直线对位函数 1.....	22
2.17.3 4 相机直线标定函数 2.....	22
2.17.4 4 相机直线对位函数 2.....	23
2.17.5 4 相机直线标定函数 3.....	23
2.17.6 4 相机直线对位函数 3.....	24

2.18	5 相机直线对位.....	24
2.18.1	5 相机直线标定函数 1.....	24
2.18.2	5 相机直线对位函数 1.....	25
2.18.3	5 相机直线标定函数 2.....	25
2.18.4	5 相机直线对位函数 2.....	26
2.19	6 相机直线对位.....	26
2.19.1	6 相机直线标定函数 1.....	26
2.19.2	6 相机直线对位函数 1.....	27
2.19.3	6 相机直线标定函数 2.....	28
2.19.4	6 相机直线对位函数 2.....	28
2.20	像素、平台坐标转换函数.....	29
2.21	像素点镜像函数.....	29
2.22	特殊对位函数.....	29
2.22.1	同边两点对位应用, 不需要标定参数.....	29
2.22.2	同边两点对位应用, 需要标定参数.....	30
2.22.3	对角对位应用, 不需要标定参数.....	30
2.22.4	对角对位应用, 需要标定参数.....	31
2.22.5	普通 3 相机对位, 需要标定参数.....	31
2.22.6	普通 4 相机对位, 需要标定参数.....	32
2.22.7	曝光机 3 相机对位, 需要标定参数.....	32
2.22.8	曝光机 4 相机对位, 需要标定参数.....	33
第 3 章	重要的使用说明.....	34
3.1	相机映射说明.....	34
3.2	关于 b_CW、b_Y 参数.....	36
3.3	实用的结构体变量参数.....	39
3.3.1	运动轴信息结构变量.....	39
3.3.2	运动平台类型.....	41
3.3.3	脉冲结构变量.....	41
3.3.4	五点标定坐标结构变量.....	42
3.3.5	对位结果结构变量.....	43
3.3.6	标定信息结构变量.....	43
3.3.7	标定直线结构变量与直线标定结果结构变量.....	44
3.3.8	自标定结果结构变量.....	44
3.4	关于 XXY、XYY 与 XXYY 平台.....	45

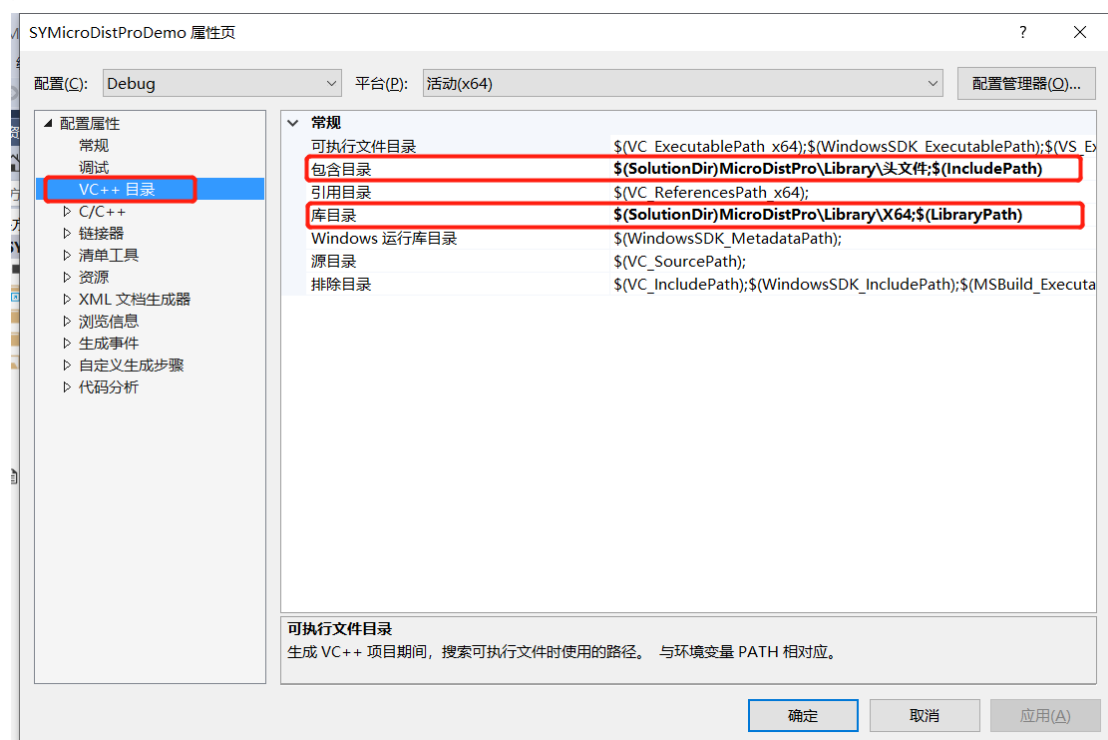
第 1 章 开发说明

1.1 函数库使用

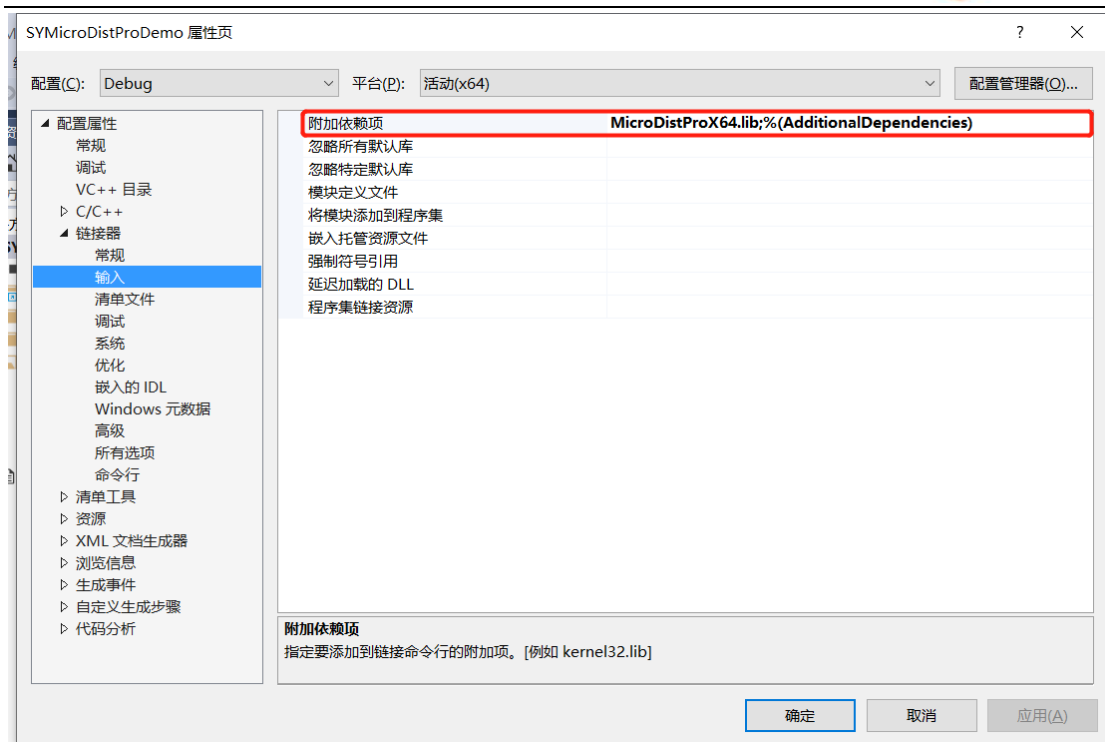
在 Windows 系统下，用户可以使用支持动态链接库的开发工具来开发应用程序。目前图像模块只支持 Visual Studio 2010 开发环境。

使用 Visual Studio 2010 来开发图像处理流程：

- 【1】 启动 Visual Studio 2010， 新建一个工程；
- 【2】 将安装目录的 Library 文件夹内 dll 文件、h 文件和 lib 文件复制到工程文件夹中；
- 【3】 选择“Project” 菜单下的“Properties...” 菜单项；
- 【4】 选择“VC++目录”， 设置包含目录为.h 文件的目录位置， 库目录为.lib 文件的目录位置。



- 【5】 切换到“链接器/输入” 标签页， 在“附加依赖项” 栏中输入 lib 文件名 MicroDistProX64.lib（32 位使用 MicroDistProX86.lib）， 如下图所示。



【6】 在应用程序文件中加入函数库头文件的声明， #include “MicroDistPro.h”；至此，用户就可以在Visual C++中调用库中的函数开发定位应用程序。如下图所示。

```

// SYMicroDistProDemoDlg.h : header file
//
#include "MicroDistPro.h"
#include "SYdef.h"
#include "SYIntegrationClass.h"
#pragma once
    
```

【注意】另外，请将hast_rt.exe放在应用程序.exe同一目录下，否则会报0XC000044错误。

【注意】应将“MicroDistProX64.dll”（32位使用MicroDistProX86.dll）文件和建立的应用程序放在同一目录内或者系统文件目录（一般如：c:\windows\system32）下，否则会出现找不到dll文件的提示。


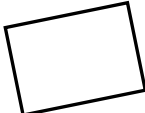

1.2 应用指南

1.2.1 对位方式

常见的对位模式有以下方式

(1)对位方式 1: 对位目标固定

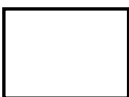
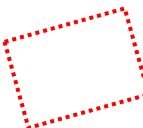
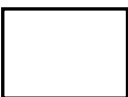

设定对位目标位置后，每次都与目标位置比较，计算出位置偏差，使对准目标位置。

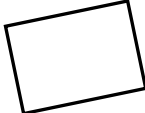
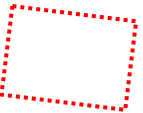
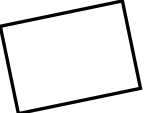

标准模板	当前位置	执行对位后
		

(2)对位方式 2: 对位目标变化

对位目标位置由产品 A 提供，而产品 A 每次位置都不一样，所以产品 B 的对位目标位置是变化的。这种方式，需要把产品 A 的目标位置通过相机映射到产品 B 坐标系中。

对位方式示意图如下：

当前位置		执行对位后	
产品 A	产品 B	产品 A	产品 B
			

当前位置		执行对位后	
产品 A	产品 B	产品 A	产品 B
			

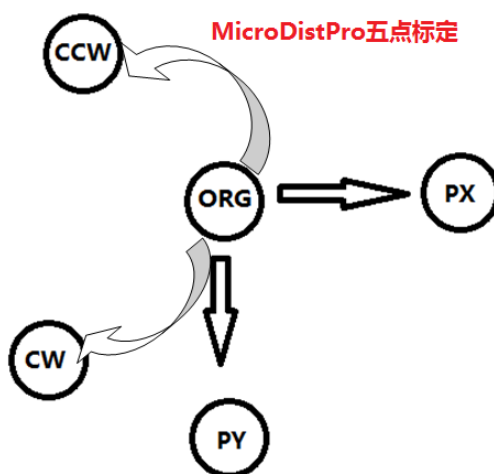
1.2.2 标定和对位流程

对位算法主要实现计算功能，根据外部提供的 5 个运动特征点完成标定；根据提供的目标位置及样本位置，计算返回偏移量。

标定和对位过程中，运动控制及图像处理是由外部完成的。

标定流程（5 步）：

- 【1】平台在基准位置时候，获取 Mark 点的像素坐标
- 【2】使平台从基准位置向+X 方向移动，获取移动后的 Mark 点的像素坐标
- 【3】使平台从基准位置向+Y 方向移动，获取移动后的 Mark 点的像素坐标
- 【4】使平台从基准位置向+ θ 方向[顺时针旋转]移动，获取移动后的 Mark 点的像素坐标
- 【5】使平台从基准位置向- θ 方向[逆时针旋转]移动，获取移动后的 Mark 点的像素坐标

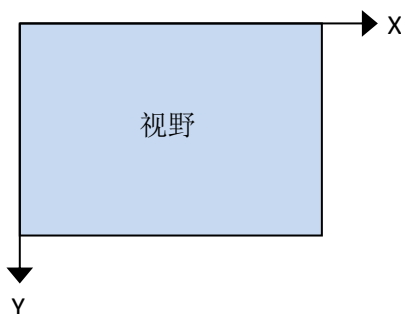


调用 `SY_MD_Free_Calibration()` 函数完成标定。

调用 `SY_MD_Free_Align()` 函数返回 X/Y/R 的位置偏移量，执行运动控制完成对位。

1.2.3 对装有棱镜装置的视觉系统

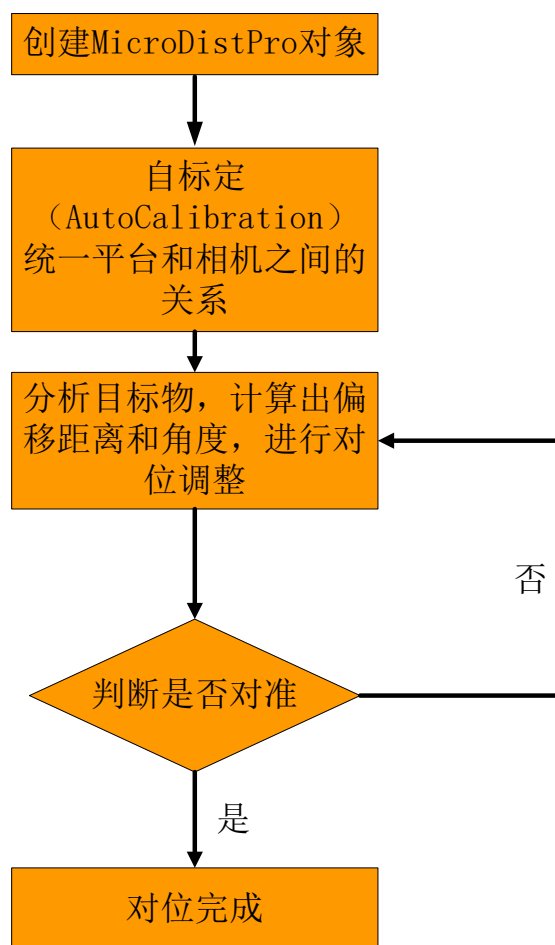
一般地，相机像素坐标系轴向关系如下图所示，即 Y 正方向在正方向右旋转 90 度方位。



MicroDistPro 算法中相机的坐标系采用此标准坐标系，所有的数字运算均在标准坐标系下进行。

由于安装需要而带有棱镜装置的视觉系统，请先对相机做 X 或 Y 方向的镜像。

1.2.4 程序流程图



第 2 章 函数说明

函数一览表		
1. SY MD GetVersion	获取版本信息	Get the MicroDist information
2. SY MD Diagonal XY	对角补偿量计算	Diagonal Compensation
3. SY MD RTCP	计算虚拟点函数	Real position transfer to pixel
4. SY MD Free Calibrate	相机标定	Free calibration
5. SY MD Free Align	2 相机对位函数	Free Align
6. SY MD Free Align2		Free Align 2 补偿各视野独立坐标系
7. SY MD Camera Info	获取标定信息	Information of the camera
8. SY MD StudyAtoB	相机映射学习函数	Cameras inline study
9. SY MD TransferAtoB	相机映射函数 A	Cameras inline study
10. SY MD TransferBtoA	相机映射函数 B	Cameras inline study
11. SY MD Nominal Dist	视野内测距函数	
12. SY MD Nominal Dist2	2 相机测距函数	
13. SY MD Physic To Pulse	物理量转换脉冲	Transfer Physic to pulse
14. SY MD Free Align3C	3 相机对位函数	Three Camera Align
15. SY MD Free Align4C	4 相机对位函数	Four Camera Align
16. SY MD Expose Align3C	3 相机的曝光机对位函数	Expose Machine 3 Camera
17. SY MD Expose Align4C	4 相机的曝光机对位函数	Expose Machine 4 Camera
18. SY MD Free Diagonal Align1	对角对位函数 1	【直角边交点】
19. SY MD Free Diagonal Align2	对角对位函数 2	【产品长宽尺寸】
20. SY MD Free Diagonal Align3	对角对位函数 3	【Org 与 Sampe 各自的标定信息】
21. SY MD SignalPoint Align	单点对位函数	单点+角度差对位
22. SY MD LineCalibration4 1	4 相机边标定	【类型 1】
23. SY MD LineAlign4 1	4 相机边对位函数	
24. SY MD LineCalibration4 2	4 相机边标定函数	【类型 2】
25. SY MD LineAlign4 2	4 相机边对位函数	
26. SY MD LineCalibration4 3	4 相机边标定函数	【类型 3】
27. SY MD LineAlign4 3	4 相机边对位函数	
28. SY MD LineCalibration5 1	5 相机边标定函数	【类型 1】
29. SY MD LineAlign5 1	5 相机边对位函数	
30. SY MD LineCalibration5 2	5 相机边标定函数	【类型 2】
31. SY MD LineAlign5 2	5 相机边对位函数	
32. SY MD LineCalibration6	6 相机边标定函数	【类型 1】
33. SY MD LineAlign6	6 相机边对位函数	
34. SY MD LineCalibration6 2	6 相机边标定函数	【类型 2】
35. SY MD LineAlign6 2	6 相机边对位函数	
36. SY MD Pixel To World	像素转换至机械平台坐标	将像素坐标转换到运动平台坐标

37. <u>SY_MD_Point_MirrorX</u>	像素点镜像函数	
38. <u>SY_MD_Point_MirrorY</u>	像素点镜像函数	
39. <u>SY_MD_Special_Align1</u>	特殊对位函数 1	同边两点对位应用, 不需要标定参数
40. <u>SY_MD_Special_Align2</u>	特殊对位函数 2	同边两点对位应用, 需要标定参数
41. <u>SY_MD_Special_Align2_1</u>	特殊对位函数 2_1	
42. <u>SY_MD_Special_Align3</u>	特殊对位函数 3	对角对位应用, 不需要标定参数
43. <u>SY_MD_Special_Align4</u>	特殊对位函数 4	对角对位应用, 需要标定参数
44. <u>SY_MD_Special_Align5</u>	特殊对位函数 5	普通 3 相机对位, 需要标定参数
45. <u>SY_MD_Special_Align6</u>	特殊对位函数 6	普通 4 相机对位, 需要标定参数
46. <u>SY_MD_Special_Align7</u>	特殊对位函数 7	曝光机 3 相机对位, 需要标定参数
47. <u>SY_MD_Special_Align8</u>	特殊对位函数 8	曝光机 4 相机对位, 需要标定参数
48. <u>SY_MD_Special_Align9</u>	特殊对位函数 9	单 Mark 对位不映射测试函数【未通过】
49. <u>SY_MD_Special_Align10</u>	特殊对位函数 10	4 点对 2 点 像素分辨率一致
50. <u>SY_MD_Special_Align11</u>	特殊对位函数 11	4 点对 2 点 像素分辨率不一致
51. <u>SY_MD_Special_Align12</u>	特殊对位函数 12	单目双 Mark, 需要标定参数
52. <u>SY_MD_Free_Align4To2</u>	4 点对 2 点对位函数	带映射的 4 点对 2 点
53. <u>SY_MD_Special_Align13</u>	特殊对位函数 13	对角对位应用, 需要标定参数
54. <u>SY_MD_Special_Align14</u>	特殊对位函数 14	对角对位应用, 需要标定参数
55. <u>SY_MD_Special_Align_TEST</u>	临时测试函数	[org 自行计算偏差]
56. <u>SY_MD_Special_Align15</u>	特殊对位函数 15	Org 为平行边, Samp 为垂直边

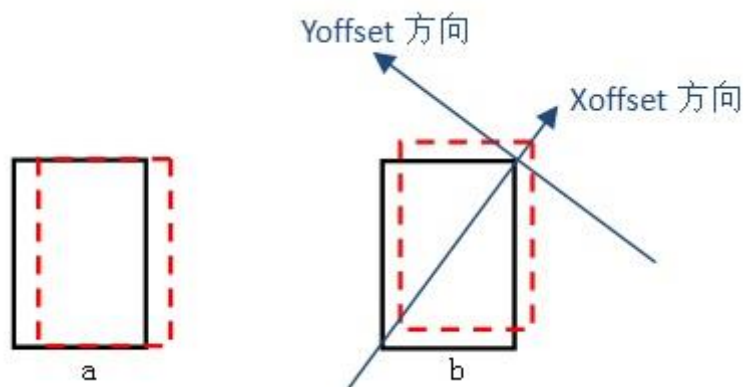
2.1 MicroDistPro 版本信息

`char*` `SY_MD_GetVersion()`

2.2 对角补偿分量计算函数

`Point2dF` `SY_MD_Diagonal_XY(float a, float b, float Dx, float Dy)`

由于 **MicroDistPro** 系统的标定特性, 会认为两特征点的连线就是像素坐标系的 X 轴, 垂直两特征点的方向为像素坐标系的 Y 轴。所以, 对于以对角特征点的对位应用, 输入的 Xoffset、Yoffset 补偿量就会作用于两对角特征点连线方向及其垂直方向上。



举例，如左图 a 是需要的产品 X 方向补偿量，由于上述的原因，补偿效果作用在对角特征点连线方向上如右图 b 所示。为了得到左图 a 的补偿效果，就需要 Xoffset 和 Yoffset 的相互作用来实现。

此函数针对对角特征点对位应用，根据产品的特征值（长度与宽度），计算出实际产品补偿量所对应的 Xoffset 和 Yoffset 的物理标量。

用户请根据设备的具体结构，改变标量的正负号甚至调换 X 和 Y 轴的值，填入对位函数中的 Xoffset 和 Yoffset 参数进行补偿。

float a、b: 产品的长度和宽度（mm）

float Dx、Dy: 产品 X、Y 方向的补偿量（mm）

注：对于特征点在产品同一边的对位应用，直接填入产品方向补偿量至对位函数中的 Xoffset 和 Yoffset 参数进行补偿，无需调用此函数。

2.3 物理量转换像素量函数

`Point2dF SY_MD_RTCP(Point5dVec CPM, Point2dF P, float Xmm, float Ymm)`

通过此函数计算出当前像素坐标点 P 移动 Xmm 和 Ymm 之后的坐标推算值，即虚拟点计算。

CPM: 相机标定信息。

P: 当前 Mark 点的像素坐标值。

Xmm、Ymm: X、Y 方向的物理距离，单位为“mm”。

2.4 自标定函数

`Point5dVec SY_MD_Free_Calibrate(`
`float m_XYdist,`
`float m_Angle,`
`Point10dF MPoint,`
`bool b_CW,`
`bool b_Y`

)

返回值为 [Point5dVec](#) 结构体，此结构体记录相机和运动平台之间的关系。

m_XYdist: X 和 Y 轴的移动量，以 Mark 不超出视野为准，单位为“mm”。

m_Angle: 旋转角度量，以 Mark 不超出视野为准，单位为 degree。

MPoint: 记录运动坐标点，见标定结构变量 [Point10dF](#) 定义

b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台 x、y 轴方向关系与相机一致，则为 true，否则为 false。

b_CW、**b_Y** 的设置详细请参见 [3.2 关于 b_CW、b_Y 参数](#)

2.5 基于坐标系进行补偿的对位函数

```
AlignData SY_MD_Free_Align(  
    Point5dVec CPMA, Point5dVec CPMB,  
    Point2dF ORGA, Point2dF ORGB,  
    Point2dF SAMPA, Point2dF SAMPB,  
    float Xoffset, float Yoffset, float Roffset,  
    bool b_CW, bool b_Y  
)
```

标准的两相机对位函数，返回原点与样本点之间的偏移量，请参见 [3.3.5 对位结果结构变量](#)。

CPMA、**CPMB**: 相机 A、B 的标定信息，通过 [2.4 自标定函数](#) 标定后得到。

ORGA、**ORGB**: 相机 A、B 的对位原点。

SAMPA、**SAMPB**: 相机 A、B 的样本点。

Xoffset、**Yoffset**: X、Y 方向的补偿值 单位 “mm”。

Roffset: 角度补偿值，单位为“度”。

b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

b_CW、**b_Y** 的设置详细请参见 [3.2 关于 b_CW、b_Y 参数](#)。

2.6 基于每个相机进行补偿的对位函数

```
AlignData SY_MD_Free_Align2(  
    Point5dVec CPMA, Point5dVec CPMB,  
    Point2dF ORGA, Point2dF ORGB,  
    Point2dF SAMPA, Point2dF SAMPB,  
    Point2dF CompensateA, Point2dF CompensateB,  
    bool b_CW, bool b_Y  
)
```

标准的两相机对位函数，返回原点与样本点之间的偏移量，请参见 [3.3.5 对位结果结构变量](#)。

CPMA、**CPMB**: 相机 A、B 的标定信息，通过 [2.1 自标定函数](#) 标定后得到。

ORGA、ORGB: 相机 A、B 的对位原点。

SAMPA、SAMPB: 相机 A、B 的样本点。

CompensateA: 相机 A 的补偿量 (x, y)。

CompenSateB: 相机 B 的补偿量 (x, y)。

b_CW: 发正脉冲时, 运动平台顺时针旋转, 则为 true, 否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致, 则为 true, 否则为 false。

b_CW、**b_Y** 的设置详细请参见 [3.2 关于 b_CW、b_Y 参数](#)。

2.7 标定信息函数

`CamInfo` `SY_MD_Camera_Info(int ImgWidth, int ImgHeight, Point5dVec CPM)`

返回值为 `CamInfo` 结构体, 可以得到相机的视野大小和像素精度。

ImgWidth: 图像宽度。

ImgHeight: 图像高度。

CPM: 相机标定信息, 通过 [2.4 自标定函数](#) 标定后得到。

2.8 相机映射学习相关函数

2.8.1 相机映射学习函数

`TransferParam` `SY_MD_StudyAtoB(Point2dF A[2], Point2dF B[2], bool CXdirRev, bool CYdirRev)`

学习两相机间的数据映射关系, 对于 [1.2.1 对位方式](#) 中提到方式 2 的对位形式, 需要将对位原点映射至对位平台坐标上。

A[2]: 相机A获取的两点坐标

B[2]: 相机B获取的两点坐标

CXdirRev、CYdirRev: A、B相机坐标轴的关系, 详细请参见 [3.1相机映射说明](#)。

2.8.2 相机映射函数 A

`Point2dF` `SY_MD_TransferAtoB(Point2dF A, TransferParam TP, bool CXdirRev, bool CYdirRev)`

把相机A中的点坐标映射到相机B坐标系中。

A: 相机A获取的点坐标

TP: `SY_MD_StudyAtoB` 相机学习时候返回的结构体, 此结构体记录相机和相机之前的关系。

CXdirRev、CYdirRev: A、B相机坐标轴的关系, 详细请参见 [3.1相机映射说明](#)。

2.8.3 相机映射函数 B

`Point2dF` `SY_MD_TransferBtoA(Point2dF B, TransferParam TP, bool CXdirRev, bool CYdirRev)`

把相机 B 中的点坐标映射到相机 A 坐标系中。

B: 相机B获取的点坐标

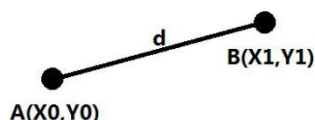
TP: `SY_MD_StudyAtoB`相机学习时候返回的结构体，此结构体记录相机和相机之前的关系。

CXdirRev、CYdirRev: A、B相机坐标轴的关系，详细请参见 [3.1相机映射说明](#)。

2.9 测距函数

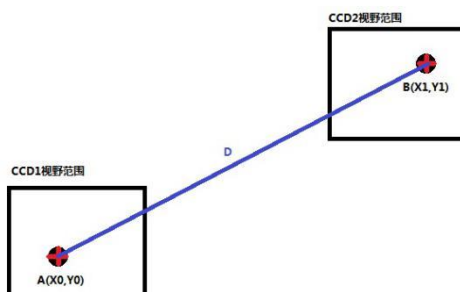
为了应用方便，我们设计出测量两点之间的距离函数。

(1) 测量同一个视野范围内两点之间的距离



如上图所示，若是视野范围内有 A 和 B 两点，且知道两个点之间的像素坐标，那么我们可通过函数 `SY_MD_Nominal_Dist` 得到 A 和 B 两点之间的物理距离。

(2) 测量不同视野范围内两点之间的距离



如上图所示，若是不同的相机的视野范围内各自有一个坐标点为 A 和 B，我们要求出两个坐标点 A 和 B 之间的物理距离，那么我们可通过函数 `SY_MD_Nominal_Dist2` 来实现。

注明：

需要特别注意的地方，我们在使用 `SY_MD_Nominal_Dist2` 函数来求得的物理距离，由于存在标定的误差，并不能反应真实的物理距离。但在一些特殊的场合，我们仅需要通过此函数求得两个位置之间的相对关系，则可通过此方法来实现。

2.9.1 图像通道测距函数

`double` `SY_MD_Nominal_Dist(Point5dVec CPM, Point2dF PointA, Point2dF PointB)`

测量同一视野内两点间的物理距离，返回值单位为“mm”。

CPM: 相机标定信息。

PointA: 相机视野内 A 点的像素坐标。

PointB: 相机视野内 B 点的像素坐标。

2.9.2 图像通道测距函数 2

```
double SY_MD_Nominal_Dist2(  
    Point5dVec CPMA, Point5dVec CPMB,  
    Point2dF PointA, Point2dF PointB,  
    bool b_Y  
)
```

通过两相机测量两 Mark 点之间的物理距离，返回值单位为“mm”。

CPMA、CPMB: 相机 A、B 标定信息，[2.4 自标定函数](#) 标定后得到。

PointA: 相机视野范围内 A 点的像素坐标。

PointB: 相机视野范围内 B 点的像素坐标。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

详细参见 [3.2 关于 b CW、b Y 参数](#)

2.10 物理量转换为脉冲量

```
PULSE SY_MD_Physic_To_Pulse(  
    Stage Table,  
    AxisInfo XDir_Axis, AxisInfo YDir_Axis, AxisInfo RDir_Axis,  
    float Xmm, float Ymm, float Degree  
)
```

将物理量转换为对应平台各轴的脉冲量，请参见 [3.3.3 脉冲结构变量](#)。

Table: 选择平台类型，包括 XYR，XXY，XYX 和 XXYY 等平台类型。

XDir_Axis 和 **YDir_Axis:**

表示 X 方向和 Y 方向的某个轴信息。

XYR 平台，则填入 X 和 Y 的轴信息。

XXY 平台，则填入 X1 和 Y 的轴信息（或 X2 和 Y 的轴信息）。

XXYY 平台，则走入 X1 和 Y1 的轴信息（或者任意 X 方向和 Y 方向的轴信息组合）。

RDir_Axis: 旋转轴信息【XXY、XXYY 平台时建立一个空变量作为参数即可】。

Xmm、Ymm: X、Y 方向的物理量，单位为“mm”。

Degree: 旋转方向的物理量，单位为“度”。

2.11 3 相机对位函数

```
AlignData SY_MD_Free_Align3C(  
    Point5dVec CPMA, Point5dVec CPMB, Point5dVec CPMC,  
    Point2dF ORGA, Point2dF ORGB, Point2dF ORGC,
```



```
Point2dF SAMPA, Point2dF SAMPB, Point2dF SAMPC,
float Xoffset, float Yoffset, float Roffset,
bool b_CW, bool b_Y, int AngleFlag
)
```

3 相机的对位函数，返回对位原点与样本点之间的偏移量，请参见 [3.3.5 对位结果结构变量](#)。

CPMA、CPMB、CPMC: 相机 A、B、C 的标定信息，通过 [2.4 自标定函数](#) 标定后得到。

ORGA、ORGB、ORGC: 相机 A、B、C、D 的对位原点。

SAMPA、SAMPB、SAMPC: 相机 A、B、C、D 的样本点。

Xoffset、Yoffset: X、Y 方向补偿值 单位 “mm”。

Roffset: 角度补偿值，单位为“度”。

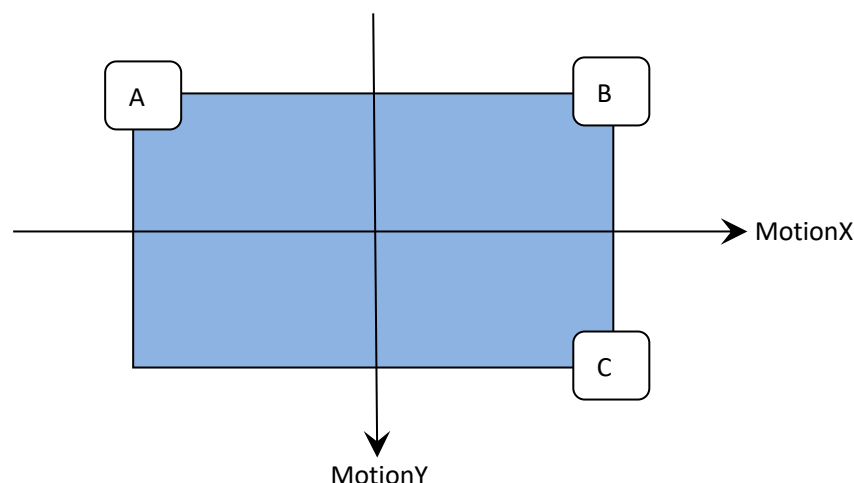
b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

AngleFlag: 角度计算方式 0: [0]AB、BC 同时约束 [1]AB 约束 [2]BC 约束

b_CW、**b_Y** 的设置详细请参见 [3.2 关于 b_CW、b_Y 参数](#)。

A、B 和 C 三个通道的位置需要形成顺时针方向，算法默认由 A 指向 B 为 X 方向，B 指向 C 为 Y 方向。偏差量和补偿量的方向均由此方向关系决定。



2.12 4 相机对位函数

```
AlignData SY_MD_Free_Align4C(
Point5dVec CPMA, Point5dVec CPMB, Point5dVec CPMC, Point5dVec CPMD,
Point2dF ORGA, Point2dF ORGB, Point2dF ORGC, Point2dF ORGD,
Point2dF SAMPA, Point2dF SAMPB, Point2dF SAMPC, Point2dF SAMPD,
float Xoffset, float Yoffset, float Roffset,
bool b_CW, bool b_Y,
int AFlag, int XYFlag,
float AngleX, float AngleY,
```

bool TF

)

4相机的对位函数，返回对位原点与样本点之间的偏移量，请参见[3.3.5 对位结果结构变量](#)。

CPMA、**CPMB**、**CPMC**、**CPMD**: 相机 A、B、C、D 的标定信息，通过 [2.4 自标定函数](#) 标定后得到。

ORGA、**ORGB**、**ORGC**、**ORGD**: 相机 A、B、C、D 的对位原点

SAMPA、**SAMPB**、**SAMPC**、**SAMPD**: 相机 A、B、C、D 的样本点

Xoffset、**Yoffset**: X 方向补偿值，单位为“mm”。

Roffset: 角度补偿值，单位为“度”。

b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

AFlag: 0--对角线角度 1--水平线角度 2--垂直线角度 3--AB 边角度 4--DC 边角度 5--AD 边角度
6--BC 边角度

XYFlag: 0--4 通道中心 1--A 坐标 2--B 坐标 3--C 坐标 4--D 坐标 5--AB 连线 6--DC 连线 7--AD 连线
8--BC 连线

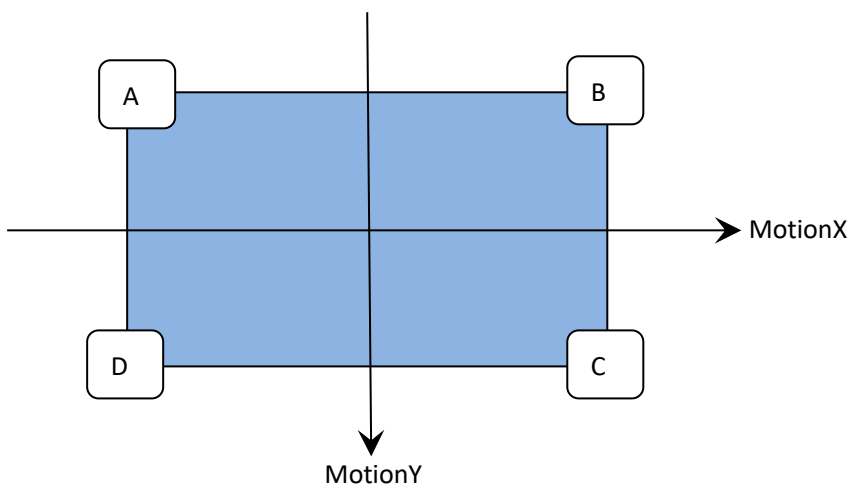
AngleX: 保留参数，默认为 0。

AngleY: 保留参数，默认为 0。

TF: 保留参数，默认为 false。

b_CW、**b_Y** 的设置详细请参见 [3.2 关于 b_CW、b_Y 参数](#)。

A、B、C 和 D 四个通道的位置需要形成顺时针方向，算法默认由 (A 指向 B)、(D 指向 C) 为 X 方向；(A 指向 D)、(B 指向 C) 为 Y 方向。偏差量和补偿量的方向均由此方向关系决定。



2.13 3 相机曝光机对位函数

AlignData

SY_MD_Expose_Align3C(

Point5dVec CPMA, Point5dVec CPMB, Point5dVec CPMC,

```
Point2dF ORGA, Point2dF ORGB, Point2dF ORGC,  
Point2dF SAMPA, Point2dF SAMPB, Point2dF SAMPC,  
bool b_CW, bool b_Y  
)
```

该函数采用 MinMax 方法进行偏移量计算。

CPMA、CPMB、CPMC: 相机 A、B、C 的标定信息

ORGA、ORGB、ORGC: 相机 A、B、C 的对位原点

SAMPA、SAMPB、SAMPC: 相机 A、B、C 的样本点

b_CW: 发正脉冲时, 运动平台顺时针旋转, 则为 true, 否则为 false

b_Y: 运动平台中 x、y 轴方向关系与相机一致, 则为 true, 否则为 false

2.14 4 相机曝光机对位函数

```
AlignData SY_MD_Expose_Align4C(  
    Point5dVec CPMA, Point5dVec CPMB, Point5dVec CPMC, Point5dVec CPMD,  
    Point2dF ORGA, Point2dF ORGB, Point2dF ORGC, Point2dF ORGD,  
    Point2dF SAMPA, Point2dF SAMPB, Point2dF SAMPC, Point2dF SAMPD,  
    bool b_CW, bool b_Y  
)
```

该函数采用 MinMax 方法进行偏移量计算。

CPMA、CPMB、CPMC、CPMD: 相机 A、B、C、D 的标定信息

ORGA、ORGB、ORGC、ORGD: 相机 A、B、C、D 的对位原点

SAMPA、SAMPB、SAMPC、SAMPD: 相机 A、B、C、D 的样本点

b_CW: 发正脉冲时, 运动平台顺时针旋转, 则为 true, 否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致, 则为 true, 否则为 false。

2.15 对角对位函数

2.15.1 对角对位函数 1

```
AlignData SY_MD_Free_Diagonal_Align1(  
    Point5dVec CPMAT, Point5dVec CPMBT,  
    Point2dF ORGA, Point2dF ORGB,  
    Point2dF SAMPA, Point2dF SAMPB,  
    LineData OrgLineA[2], LineData OrgLineB[2],  
    LineData SampLineA[2], LineData SampLineB[2],  
    float Xoffset, float Yoffset, float Angleoffset,  
    bool b_CW, bool b_Y)
```

此函数适用于图像处理方法为求直线交点的对角对位应用。

CPMAT、CPMBT: 相机 A、B 的标定信息

ORGA、ORGB: 相机 A、B 的对位原点

SAMPA、SAMPB: 相机 A、B 的样本点

OrgLineA[2]、OrgLineB[2]: 相机 A、B 中原始点直线

SampLineA[2]、SampLineB[2]: 相机 A、B 中样本点直线

Xoffset、Yoffset: X 方向补偿值 单位 mm

Angleoffset: 角度补偿值, 单位为“度”

b_CW: 发正脉冲时, 运动平台顺时针旋转, 则为 true, 否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致, 则为 true, 否则为 false。

2.15.1 对角对位函数 2

```
AlignData          SY_MD_Free_Diagonal_Align2(  
    Point5dVec CPMAT, Point5dVec CPMBT,  
    Point2dF ORGA, Point2dF ORGB,  
    Point2dF SAMPA, Point2dF SAMPB,  
    float OrgLength, float OrgWidth,  
    float SampLength, float SampWidth,  
    float Xoffset, float Yoffset, float Angleoffset,  
    bool b_CW, bool b_Y  
)
```

该函数适用于能提供产品尺寸的对角对位应用。

CPMAT、CPMBT: 相机 A、B 的标定信息。

ORGA、ORGB: 相机 A、B 的对位原点。

SAMPA、SAMPB: 相机 A、B 的样本点。

OrgLength、OrgWidth: ORG 产品的长宽。

SampLength、SampWidth: SAMP 产品的长宽。

Xoffset、Yoffset: X 方向补偿值 单位为“mm”。

Angleoffset: 角度补偿值, 单位为“度”。

b_CW: 发正脉冲时, 运动平台顺时针旋转, 则为 true, 否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致, 则为 true, 否则为 false。

2.15.3 对角对位函数 3

```
AlignData          SY_MD_Free_Diagonal_Align3(  
    Point5dVec OrgACPM, Point5dVec OrgBCPM,  
    Point5dVec SampACPM, Point5dVec SampBCPM,  
    Point2dF ORGA, Point2dF ORGB,  
    Point2dF SAMPA, Point2dF SAMPB,  
    float OrgLength, float OrgWidth,  
    float SampLength, float SampWidth,
```

```
float Xoffset, float Yoffset, float Angleoffset,  
bool b_CW, bool b_Y  
)
```

该函数适用于能提供产品尺寸，且 ORG 相机与 SAMPLE 相机不相同的对角对位应用。

OrgACPM、OrgBCPM: ORG 相机 A、B 的标定信息。

SampACPM、SampBCPM: SAMPLE 相机 A、B 的标定信息。

ORGA、ORGB: 相机 A、B 的对位原点。

SAMPA、SAMPB: 相机 A、B 的样本点。

OrgLength、OrgWidth: ORG 产品的长宽。

SampLength、SampWidth: SAMP 产品的长宽。

Xoffset、Yoffset: X 方向补偿值 单位为“mm”。

Angleoffset: 角度补偿值，单位为“度”。

b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

2.16 单点对位函数

```
AlignData SY_MD_SignalPoint_Align(  
    Point5dVec CPM,  
    Point2dF org, Point2dF samp,  
    float OrgAngle, float DeltaAngle,  
    bool b_CW, bool b_Y,  
    float Xoffset, float Yoffset, float Roffset  
)
```

该函数适用于 Org、Samp 产品都能提供各自 X、Y、Angle 的对位应用。

CPM: 标定参数

Org、Samp: 基准点与样本点。

OrgAngle: 基准角度。

DeltaAngle: 角度偏差，即： $(OrgAngle - SampAngle)$ Org 与 Samp 的角度差。

b_CW: 发正脉冲时，运动平台顺时针旋转，则为 true，否则为 false。

b_Y: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false。

Xoffset、Yoffset: XY 方向补偿量，单位为“mm”。

Roffset: 角度补偿量，单位为“度”。

2.17 4 相机直线对位

2.17.1 4 相机直线标定函数 1

```
LineCalibration SY_MD_LineCalibration4_1(  

```

```

Point2dF ImageSize[4],
CalibrationLines Line[4] ,
float XYDist, float RAngle
)

```

根据直线特征进行标定。

ImageSize[4]: 相机的图像分辨率

Line[4]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长, 单位为“mm”。

RAngle: 标定角度, 单位为“度”。

2.17.2 4 相机直线对位函数 1

```

AlignData      SY_MD_LineAlign4_1(
                Point2dF ImageSize[4],
                LineCalibration LC,
                LineData OrgLine[4],
                LineData SampLine[4],
                float ComX, float ComY1, float ComY2
                )

```

ImageSize[4]: 相机的图像分辨率

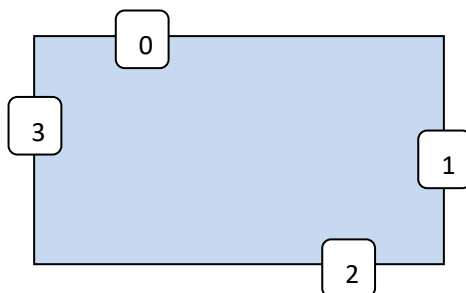
LC: 直线标定参数, 通过 [SY MD LineCalibration4_1](#) 函数来得到。

OrgLine[4]、**SampLine[4]**: 基准直线与样本直线。

ComX: 相机 2、4 方向补偿量, 单位为“mm”。

ComY1: 相机 0 方向补偿量, 单位为“mm”。

ComY2: 相机 1 方向补偿量, 单位为“mm”。



2.17.3 4 相机直线标定函数 2

```

LineCalibration SY_MD_LineCalibration4_2(
                Point2dF ImageSize[4],
                CalibrationLines Line[4] ,
                float XYDist, float RAngle

```

)

根据直线特征进行标定。

ImageSize[4]: 相机的图像分辨率

Line[4]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长, 单位为“mm”。

RAngle: 标定角度, 单位为“度”。

2.17.4 4 相机直线对位函数 2

```
AlignData      SY_MD_LineAlign4_2(
                Point2dF ImageSize[4],
                LineCalibration *LC,
                LineData OrgLine[4],
                LineData SampLine[4],
                float ComX, float ComY1, float ComY2
                )
```

ImageSize[4]: 相机的图像分辨率

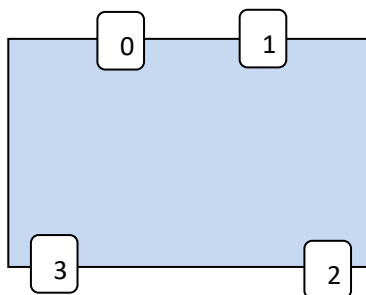
LC: 直线标定参数, 通过 [SY_MD_LineCalibration4_2](#) 函数来得到。

OrgLine[4]、**SampLine[4]**: 基准直线与样本直线。

ComX: 相机 0、1 方向补偿量, 单位为“mm”。

ComY1: 相机 2 方向补偿量, 单位为“mm”。

ComY2: 相机 3 方向补偿量, 单位为“mm”。



2.17.5 4 相机直线标定函数 3

```
LineCalibration SY_MD_LineCalibration4_3(
                Point2dF ImageSize[4],
                Point2dF Pts[4][5] ,
                float XYDist, float RAngle
                )
```

根据直线特征进行标定。

ImageSize[4]: 相机的图像分辨率

Pts[4][5]: 每个相机的五点标定的坐标信息。

XYDist: XY 标定步长，单位为“mm”。

RAngle: 标定角度，单位为“度”。

2.17.6 4 相机直线对位函数 3

```
AlignData      SY_MD_LineAlign4_3(  
                Point2dF ImageSize[4],  
                LineCalibration *LC,  
                Point2dF Org[4],  
                Point2dF Samp[4],  
                float ComX, float ComY1, float ComY2  
                )
```

ImageSize[4]: 相机的图像分辨率

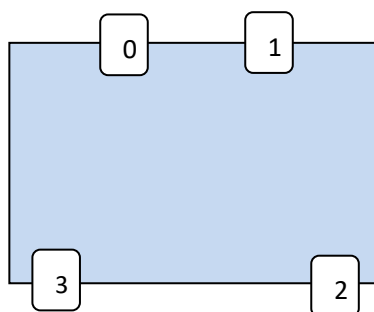
LC: 直线标定参数，通过 [SY MD LineCalibration4 3](#) 函数来得到。

Org[4]、**Samp[4]**: 每个相机中的基准点和样本点坐标。

ComX: 相机 0、1 方向补偿量，单位为“mm”。

ComY1: 相机 2 方向补偿量，单位为“mm”。

ComY2: 相机 3 方向补偿量，单位为“mm”。



2.18 5 相机直线对位

2.18.1 5 相机直线标定函数 1

```
LineCalibration SY_MD_LineCalibration5_1(  
                Point2dF ImageSize[5],  
                CalibrationLines Line[5] ,  
                float XYDist, float RAngle  
                )
```

根据直线特征进行标定。

ImageSize[5]: 相机的图像分辨率

Line[5]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长，单位为“mm”。

RAngle: 标定角度，单位为“度”。

2.18.2 5 相机直线对位函数 1

```
AlignData      SY_MD_LineAlign5_1(  
                Point2dF ImageSize[5],  
                LineCalibration LC,  
                LineData OrgLine[5],  
                LineData SampLine[5],  
                float ComX, float ComY1, float ComY2  
                )
```

该函数适用于相机 0、1 检测水平直线边的直线对位应用，如下图所示。

ImageSize[5]: 相机的图像分辨率

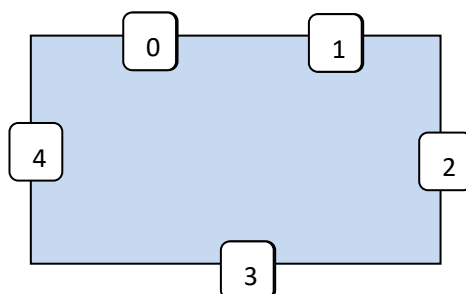
LC: 直线标定参数，通过 [SY MD LineCalibration5_1](#) 函数来得到。

OrgLine[5]、**SampLine[5]**: 基准直线与样本直线。

ComX: 相机 2、4 方向补偿量，单位为“mm”。

ComY1: 相机 0 方向补偿量，单位为“mm”。

ComY2: 相机 1 方向补偿量，单位为“mm”。



2.18.3 5 相机直线标定函数 2

```
LineCalibration SY_MD_LineCalibration5_2(  
                Point2dF ImageSize[5],  
                CalibrationLines Line[5] ,  
                float XYDist, float RAngle  
                )
```

根据直线特征进行标定。

ImageSize[5]: 相机的图像分辨率

Line[5]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长，单位为“mm”。

RAngle: 标定角度，单位为“度”。

2.18.4 5 相机直线对位函数 2

```
AlignData          SY_MD_LineAlign5_2(  
                    Point2dF ImageSize[5],  
                    LineCalibration LC,  
                    LineData OrgLine[5],  
                    LineData SampLine[5],  
                    float ComX1, float ComX2, float ComY  
                    )
```

该函数适用于相机 0、1 检验垂直边的直线对位应用，如下图所示。

ImageSize[5]:相机的图像分辨率

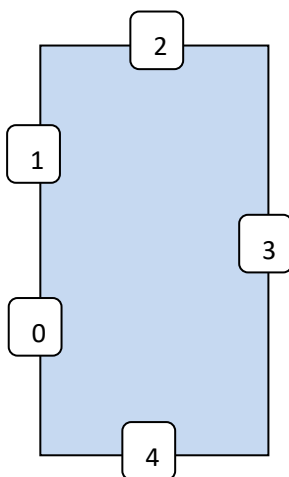
LC: 直线标定参数，通过 [SY_MD_LineCalibration5_2](#) 函数来得到。

OrgLine[5]、SampLine[5]: 基准直线与样本直线。

ComX1: 相机 2、4 方向补偿量，单位为“mm”。

ComX2: 相机 0 方向补偿量，单位为“mm”。

ComY: 相机 1 方向补偿量，单位为“mm”。



2.19 6 相机直线对位

2.19.1 6 相机直线标定函数 1

```
LineCalibration    SY_MD_LineCalibration6(  
                    Point2dF ImageSize[6],
```

```
CalibrationLines Line[6] ,  
float XYDist, float RAngle  
)
```

根据直线特征进行标定。

ImageSize[6]: 相机的图像分辨率

Line[6]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长, 单位为“mm”。

RAngle: 标定角度, 单位为“度”。

2.19.2 6 相机直线对位函数 1

```
AlignData      SY_MD_LineAlign6(  
    Point2dF ImageSize[6],  
    LineCalibration LC,  
    LineData OrgLine[6],  
    LineData SampLine[6],  
    float ComX, float ComY1, float ComY2  
)
```

ImageSize[6]: 相机的图像分辨率

LC: 直线标定参数, 通过 [SY MD LineCalibration6](#) 函数来得到。

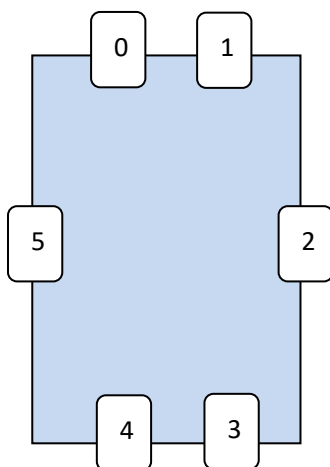
OrgLine[6]、**SampLine[6]**: 基准直线与样本直线。

ComX: 相机 2、5 方向补偿量, 单位为“mm”。

ComY1: 相机 0、4 方向补偿量, 单位为“mm”。

ComY2: 相机 1、3 方向补偿量, 单位为“mm”。

该函数适用于以下对位应用, 如下图所示。



2.19.3 6 相机直线标定函数 2

```
LineCalibration SY_MD_LineCalibration6_2(
    Point2dF ImageSize[6],
    CalibrationLines Line[6] ,
    float XYDist, float RAngle
)
```

根据直线特征进行标定。

ImageSize[6]: 相机的图像分辨率

Line[6]: 直线标定的 5 条特征直线。

XYDist: XY 标定步长, 单位为“mm”。

RAngle: 标定角度, 单位为“度”。

2.19.4 6 相机直线对位函数 2

```
AlignData SY_MD_LineAlign6(
    Point2dF ImageSize[6],
    LineCalibration LC,
    LineData OrgLine[6],
    LineData SampLine[6],
    float ComX, float ComY1, float ComY2,
    unsigned int flag
)
```

ImageSize[6]: 相机的图像分辨率

LC: 直线标定参数, 通过 [SY_MD_LineCalibration6_2](#) 函数来得到。

OrgLine[6]、**SampLine[6]**: 基准直线与样本直线。

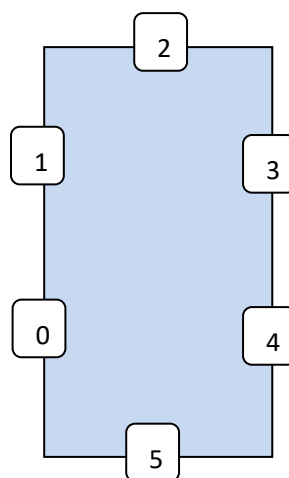
ComX: 相机 2、4 方向补偿量, 单位为“mm”。

ComY1: 相机 0 方向补偿量, 单位为“mm”。

ComY2: 相机 1 方向补偿量, 单位为“mm”。

flag: 0: 01 1: 34 2: 01&34

该函数适用于以下对位应用, 如右图所示。



2.20 像素、平台坐标转换函数

`Point2dF` `SY_MD_Pixel_To_World(Point2dF pixel, Point5dVec CPM, bool b_Y)`

该函数把像素坐标转换至标定的机构坐标系中。

`pixel`: 相机中像素坐标

`CPM`: 标定参数

`b_Y`: 运动平台中 x、y 轴方向关系与相机一致，则为 true，否则为 false

2.21 像素点镜像函数

`Point2dF` `SY_MD_Point_MirrorX(int ImgWidht, int ImgHeight, Point2dF p);`

`Point2dF` `SY_MD_Point_MirrorY(int ImgWidht, int ImgHeight, Point2dF p);`

该函数把像素坐标转换至 X 或者 Y 方向镜像后的坐标。

`ImgWidht`: 图片宽度

`ImgHeight`: 图片高度

`p`: 像素坐标

2.22 特殊对位函数

该系列函数支持不做相机映射学习或不需要做标定流程的对位应用，一般需要进行多次对位达到精度。共有参数如下表所示，私有参数见各函数注释。

<code>CamStatus</code>	对应的两相机坐标轴方向关系，两相机必须保证同轴 0: XY同方向 1: X同向Y异向 2: X异向Y同向 3: XY异方向
<code>OrgStdX</code> 、 <code>SampStdX</code>	Org产品和Samp产品的基准点
<code>OrgX</code> 、 <code>SampX</code>	Org产品和Samp产品的样本点
<code>mpp</code>	像素分辨率，mm/pixel
<code>deltaX</code> 、 <code>deltaY</code>	XY方向补偿量，单位为“mm”
<code>deltaA</code>	角度补偿量，单位为“度”。
<code>paramX</code>	相机标定参数
<code>AngleFlag</code>	[0]AB、BC同时约束 [1]AB约束 [2]BC约束 [3]视野内补偿
<code>XYFlag</code>	[0]AB中点 [1]A点 [2]B点

2.22.1 同边两点对位应用，不需要标定参数

`AlignData` `SY_MD_Special_Align1(
int CamStatus,
int mode,`

```
float OrgCamDist, float SampCamDist,  
Point2dF OrgStdA, Point2dF OrgStdB,  
Point2dF SampStdA, Point2dF SampStdB,  
Point2dF OrgA, Point2dF OrgB,  
Point2dF SampA, Point2dF SampB,  
float mpp_OrgA, float mpp_OrgB,  
float mpp_SampA, float mpp_SampB,  
float deltaX, float deltaY, float deltaA  
)
```

mode: 0—水平边两点 1—垂直边两点

OrgCamDist: Org产品长度

SampCamDist: Samp产品长度

2.22.2 同边两点对位应用，需要标定参数

```
AlignData SY_MD_Special_Align2(  
    int CamStatus,  
    Point5dVec paramA, Point5dVec paramB,  
    Point2dF OrgStdA, Point2dF OrgStdB,  
    Point2dF SampStdA, Point2dF SampStdB,  
    Point2dF OrgA, Point2dF OrgB,  
    Point2dF SampA, Point2dF SampB,  
    float mpp_OrgA, float mpp_OrgB,  
    float deltaX, float deltaY, float deltaA,  
    unsigned int XYFlag  
)
```

XYFlag: 0—AB中点 1—A点 2—B点

2.22.3 对角对位应用，不需要标定参数

该方法目前只支持图像处理方法为直线交点的应用。

```
AlignData SY_MD_Special_Align3(  
    int CamStatus,  
    float OrgCamDistW, float OrgCamDistH,  
    float SampCamDistW, float SampCamDistH,  
    Point2dF OrgStdA, Point2dF OrgStdB,  
    LineData OrgStdAngleA[2], LineData OrgStdAngleB[2],  
    Point2dF SampStdA, Point2dF SampStdB,  
    LineData SampStdAngleA[2], LineData SampStdAngleB[2],  
    Point2dF OrgA, Point2dF OrgB,
```

```
LineData OrgAngleA[2], LineData OrgAngleB[2],  
Point2dF SampA, Point2dF SampB,  
LineData SampAngleA[2], LineData SampAngleB[2],  
float mpp_OrgA, float mpp_OrgB,  
float mpp_SampA, float mpp_SampB,  
float deltaX, float deltaY, float deltaA  
)
```

OrgCamDistW、**OrgCamDistH**: Org产品的长度和宽度

SampCamDistW、**SampCamDistH**: Samp产品的长度和宽度

OrgStdAngleA[2]、**OrgStdAngleB[2]**: Org产品的基准角度

SampStdAngleA[2]、**SampStdAngleB[2]**: Samp产品的基准角度

OrgAngleA[2]、**OrgAngleB[2]**: Org产品当前角度

SampAngleA[2]、**SampAngleB[2]**: Samp产品当前角度

2.22.4 对角对位应用，需要标定参数

该方法目前只支持图像处理方法为直线交点的应用。

```
AlignData SY_MD_Special_Align4(  
    int CamStatus,  
    Point5dVec paramA, Point5dVec paramB,  
    Point2dF OrgStdA, Point2dF OrgStdB,  
    LineData OrgStdAngleA[2], LineData OrgStdAngleB[2],  
    Point2dF SampStdA, Point2dF SampStdB,  
    LineData SampStaAngleA[2], LineData SampStdAngleB[2],  
    Point2dF OrgA, Point2dF OrgB,  
    LineData OrgAngleA[2], LineData OrgAngleB[2],  
    Point2dF SampA, Point2dF SampB,  
    LineData SampAngleA[2], LineData SampAngleB[2],  
    float mpp_OrgA, float mpp_OrgB,  
    float deltaX, float deltaY, float deltaA  
)
```

OrgStdAngleA[2]、**OrgStdAngleB[2]**: Org产品的基准角度

SampStdAngleA[2]、**SampStdAngleB[2]**: Samp产品的基准角度

OrgAngleA[2]、**OrgAngleB[2]**: Org产品当前角度

SampAngleA[2]、**SampAngleB[2]**: Samp产品当前角度

2.22.5 普通 3 相机对位，需要标定参数

```
AlignData SY_MD_Special_Align5(  
    int CamStatus,
```

```
Point5dVec paramA, Point5dVec paramB, Point5dVec paramC,  
Point2dF OrgStdA, Point2dF OrgStdB, Point2dF OrgStdC,  
Point2dF SampStdA, Point2dF SampStdB, Point2dF SampStdC,  
Point2dF OrgA, Point2dF OrgB, Point2dF OrgC,  
Point2dF SampA, Point2dF SampB, Point2dF SampC,  
float mpp_OrgA, float mpp_OrgB, float mpp_OrgC,  
float deltaX, float deltaY, float deltaA,  
int AngleFlag  
)
```

AngleFlag: [0]AB、BC同时约束 [1]AB约束 [2]BC约束 [3]视野内补偿

2.22.6 普通 4 相机对位，需要标定参数

```
AlignData SY_MD_Special_Align6(  
    int CamStatus,  
    Point5dVec paramA, Point5dVec paramB,  
    Point5dVec paramC, Point5dVec paramD,  
    Point2dF OrgStdA, Point2dF OrgStdB,  
    Point2dF OrgStdC, Point2dF OrgStdD,  
    Point2dF SampStdA, Point2dF SampStdB,  
    Point2dF SampStdC, Point2dF SampStdD,  
    Point2dF OrgA, Point2dF OrgB,  
    Point2dF OrgC, Point2dF OrgD,  
    Point2dF SampA, Point2dF SampB,  
    Point2dF SampC, Point2dF SampD,  
    float mpp_OrgA, float mpp_OrgB,  
    float mpp_OrgC, float mpp_OrgD,  
    float deltaX, float deltaY, float deltaA  
)
```

2.22.7 曝光机 3 相机对位，需要标定参数

```
AlignData SY_MD_Special_Align7(  
    int CamStatus,  
    Point5dVec paramA, Point5dVec paramB, Point5dVec paramC,  
    Point2dF OrgStdA, Point2dF OrgStdB, Point2dF OrgStdC,  
    Point2dF SampStdA, Point2dF SampStdB, Point2dF SampStdC,  
    Point2dF OrgA, Point2dF OrgB, Point2dF OrgC,  
    Point2dF SampA, Point2dF SampB, Point2dF SampC,
```



```
float mpp_OrgA, float mpp_OrgB, float mpp_OrgC,  
float deltaX, float deltaY, float deltaA  
)
```

2.22.8 曝光机 4 相机对位，需要标定参数

```
AlignData      SY_MD_Special_Align8(  
    int CamStatus,  
    Point5dVec paramA, Point5dVec paramB,  
    Point5dVec paramC, Point5dVec paramD,  
    Point2dF OrgStdA, Point2dF OrgStdB,  
    Point2dF OrgStdC, Point2dF OrgStdD,  
    Point2dF SampStdA, Point2dF SampStdB,  
    Point2dF SampStdC, Point2dF SampStdD,  
    Point2dF OrgA, Point2dF OrgB,  
    Point2dF OrgC, Point2dF OrgD,  
    Point2dF SampA, Point2dF SampB,  
    Point2dF SampC, Point2dF SampD,  
    float mpp_OrgA, float mpp_OrgB,  
    float mpp_OrgC, float mpp_OrgD,  
    float deltaX, float deltaY, float deltaA  
)
```

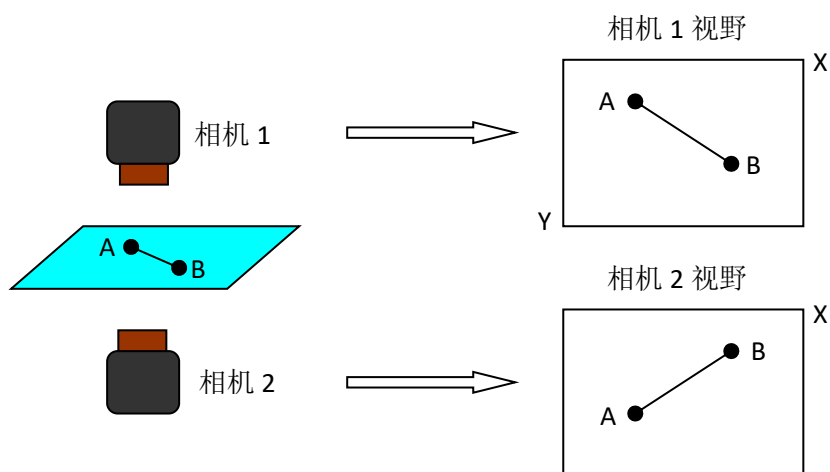
第 3 章 重要的使用说明

以上我们介绍了在 VC++ 中如何配置 **MicroDistPro** 和函数说明, 为了更好地理解 **MicroDistPro** 和正确地使用函数, 在这一节, 将论述 **MicroDistPro** 的重要概念与数据结构体。

3.1 相机映射说明

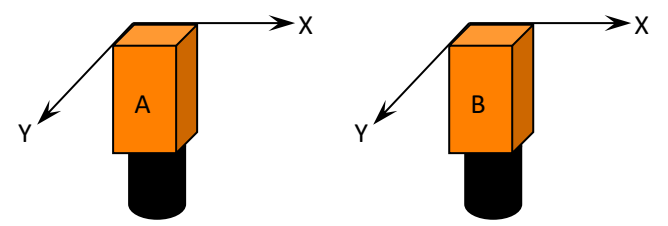
相关函数: **SY_MD_StudyAtoB()**, **SY_MD_TransferAtoB()**, **SY_MD_TransferBtoA()**。

相机映射, 是实现相机间坐标的互相转换, 以满足目标位置变化的对位模式应用。

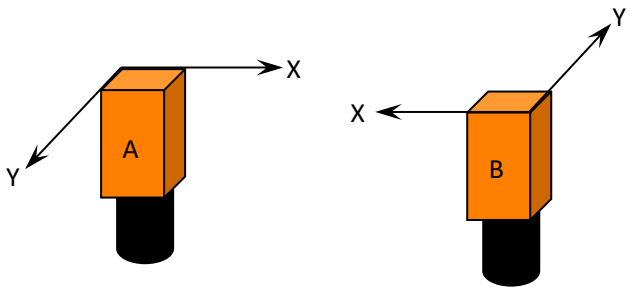


假设相机 1 和相机 2 观察同一片产品上 A、B 两点 (一般地, 产品需要从相机 1 移动至相机 2 处), 相机 1 视野中 A、B 两点坐标被保存在 Data1[2] 数组, 相机 2 视野中 A、B 两点坐标被保存在 Data2[2] 数组, 那么我们就可以调用 **SY_MD_StudyAtoB()** 实现相机映射学习。

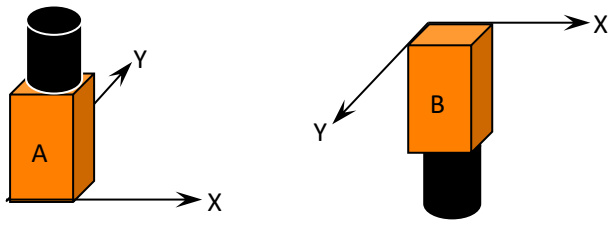
映射函数有两布尔量参数 **bool CXdirRev**、**bool CYdirRev**, 它们与映射相机的安装方式有关。为了正确使用函数, 请将相机安装成以下方式中某一种。【其它情况请垂询】

方式 1	
	CXdirRev = false CYdirRev = false

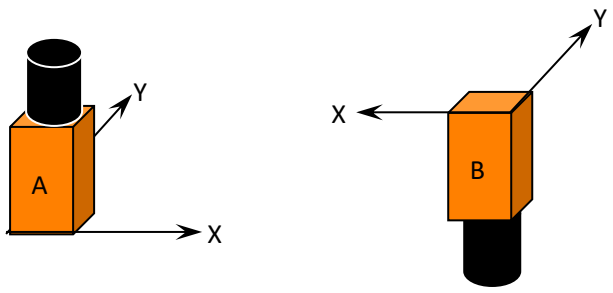
方式 2 (A、B 相机位置可以左右互换)

	<p>CXdirRev = true CYdirRev = true</p>
---	--

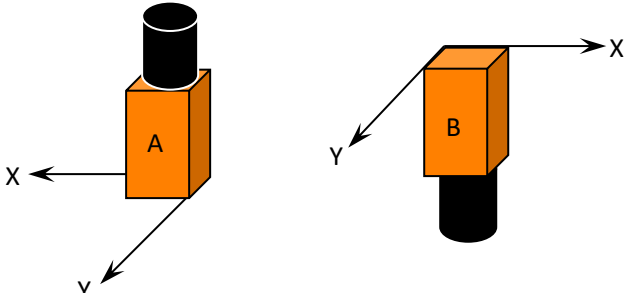
方式 3 (A、B 相机位置可以左右互换)

	<p>CXdirRev = false CYdirRev = true</p>
---	---

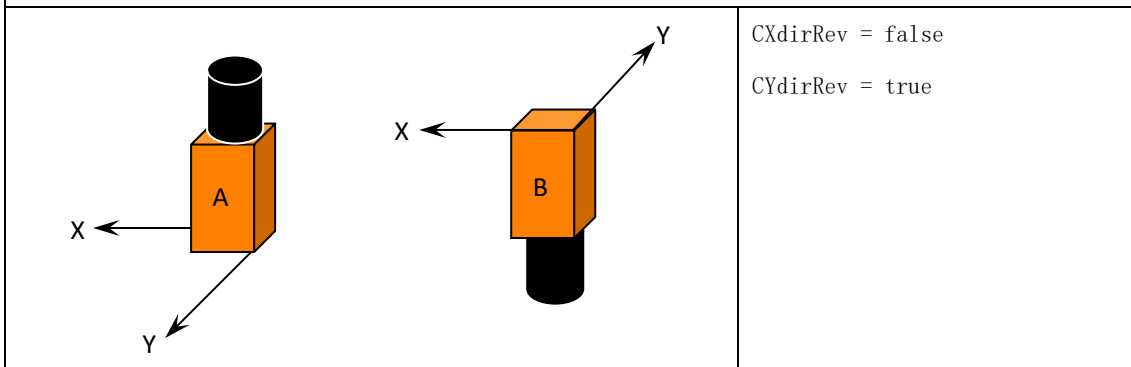
方式 4 (A、B 相机位置可以左右互换)

	<p>CXdirRev = true CYdirRev = false</p>
---	---

方式 5 (A、B 相机位置可以左右互换)

	<p>CXdirRev = true CYdirRev = false</p>
---	---

方式 6 (A、B 相机位置可以左右互换)



从以上各种方式可知，CXdirRev 和 CYdirRev 主要描述了两个相机之间像素坐标轴的方向异同性。

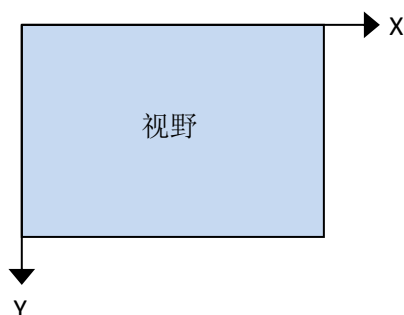
Example:

```
//相机1和相机 2参数学习转换
Point2dF A[2], B[2];
TransferParam TP12;
//图像分析，得到 A[2]、B[2]
//调用映射学习函数
TP12 = SY_MD_StudyAtoB(A,B,False,True);

//相机映射数据
Point2dF TA, TB;
//图像分析得到 TA
TB = SY_MD_TransferAtoB(TA,TP12,False,True);
```

3.2 关于 b_CW、b_Y 参数

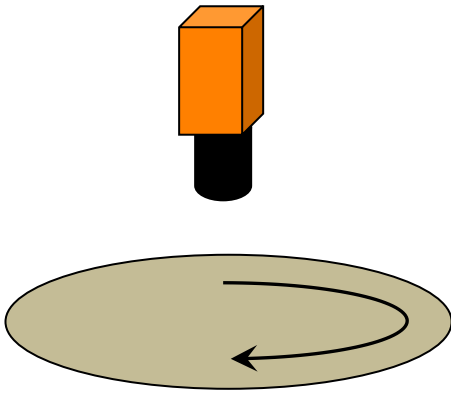
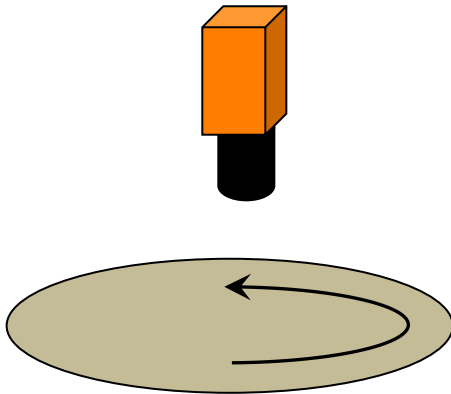
一般地，相机像素坐标系符合下图的关系：



即 Y 轴的正方向为 X 轴正方向的右旋转 90 度方位，我们的算法是基于这种标准的视觉坐标系的。

【*】某些应用的要求，相机需要透过**棱镜**取像，则这种标准坐标系被改变；此时，请做相机的 X 镜像或 Y 镜像，使改变成标准像素坐标系，方可正确地使用函数。

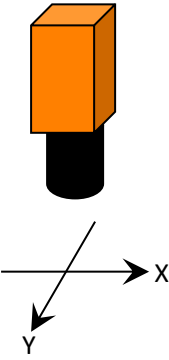
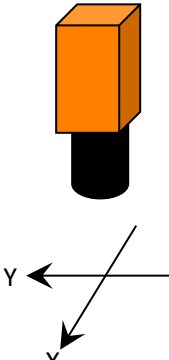
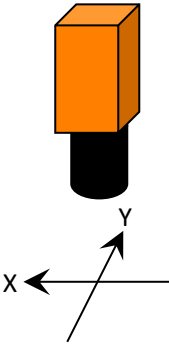
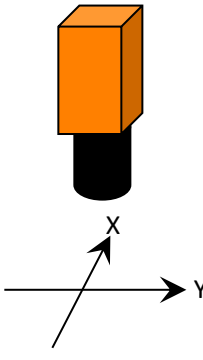
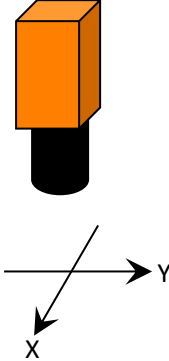
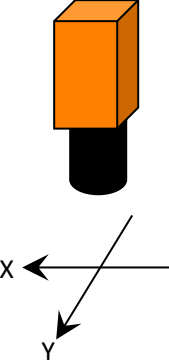
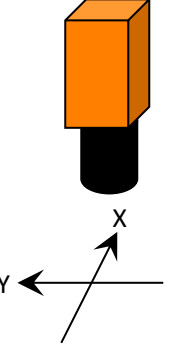
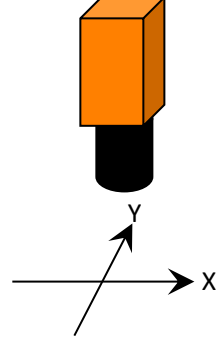
b_CW 描述了相机坐标系与机械旋转方向的关系，请参考图示进行设置。

以相机的方向观察旋转运动（相机的自旋转角度不需要考虑）	
	
当发送正脉冲时，对位平台旋转方向为顺时针，则 b_CW = true	当发送正脉冲时，对位平台旋转方向为逆时针，则 b_CW = false
对于相机由下往上观察旋转运动的情况，亦遵循这个原则。	

b_Y 描述了相机坐标系与机械 X、Y 运动轴的关系，请参考图示进行设置。

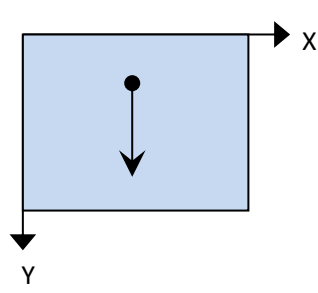
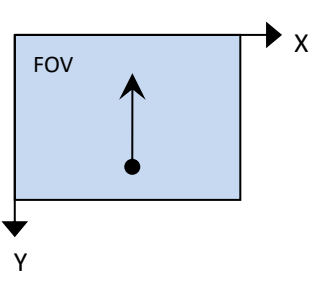
这里分两种情况，一种是相机能同时观察到 X、Y 的运动情况；另外一种情况是相机只能观察到 X 或 Y 的运动情况。

【1】以相机的方向观察平台 X、Y 运动（相机的自旋转角度不需要考虑），相机能同时观察 X、Y 方向的运动情况（标准模型）。

			
符合上面 4 种运动情况的, $b_Y = true$			
			
符合上面 4 种运动情况的, $b_Y = false$			
<p>对于相机由下往上观察 X、Y 运动的情况, 亦遵循这个原则。</p> <p>这种情况下, 对位函数返回的偏移量在数值与符号上都是正确的。</p>			

【2】对于分离式的运动平台, 相机只能观察到 X 或 Y 的运动情况(不需要考虑相机的自旋转角度), 则按下列原则设置:

相机只能观察到 X 方向运动	$b_Y = true$
----------------	--------------

相机只能观察到 Y 方向运动		
	当 Y 轴正方向运动时, 标记点在相机视野内从上往下运动 b_Y = true	当 Y 轴正方向运动时, 标记点在相机视野内从下往上运动 b_Y = false
<p>对于相机由下往上观察 X、Y 运动的情况, 亦遵循这个原则。</p> <p>这种情况下, 对位函数返回的偏移量在数值上是正确的, 但是符号需要进行测试并调整。</p>		

3.3 实用的结构体变量参数

3.3.1 运动轴信息结构变量

MicroDistPro 提供此结构来管理对位的运动轴, 是对位系统正确进行运动控制的关键。形式如下:

```
typedef struct AxisInfo
{
    float DriveRatio;
    float DriveRadius;
    float Pitch;
    float PPR;
}AxisInfo;
```

Pitch: 运动轴的螺纹距, 单位为“mm”

PPR: 运动轴电机的分辨率, 单位为 pulse

DriveRatio: 带比例的旋转轴比率, 直驱时此值为 1

DriveRadius: 带半径的旋转轴半径值, 单位为“mm”

注: **DriveRatio** 与 **DriveRadius**, 其中一种形式有效时, 另外形式的值为 0;

假设现有一带比例的旋转装置, 驱动电机分辨率 6400, 比例系数 90.0, 结构体的赋值举例如下:

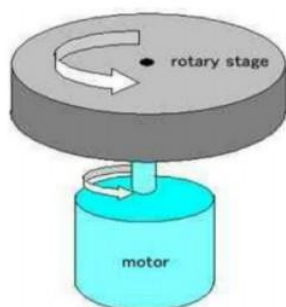
```
#define Raxis 0;
AxisInfo MotionAxisInfo[3];
```

```

MotionAxisInfo[RAxis].Pitch = 1.0;
MotionAxisInfo[RAxis].PPR = 6400;
MotionAxisInfo[RAxis].DriveRadius = 0;
MotionAxisInfo[RAxis].DriveRatio = 90.0;

```

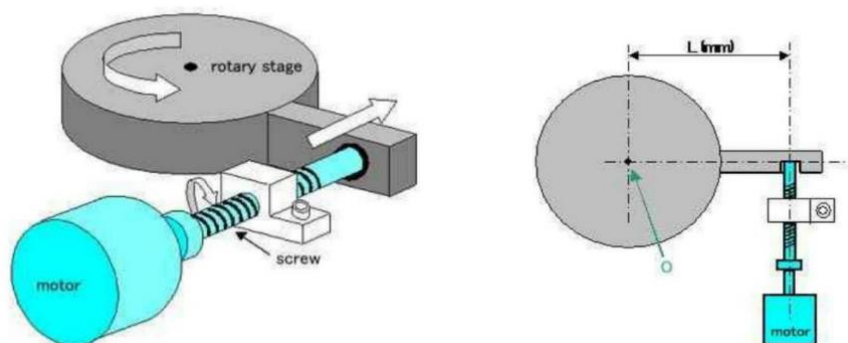
带比例的旋转运动 电机通过如蜗杆、齿轮等装置，带动做旋转运动。



假设有一比例装置，其传动比为 100，电机的分辨率为 3600；那么此旋转装置的角度分辨率：
 $360 \div 3600 \div 100 = 0.001$ （度/脉冲）

此时 DriveRatio = 100 而 DriveRadius = 0;

带半径的旋转运动 电机通过丝杆装置带动做旋转运动



此时 DriveRatio = 0 而 DriveRadius = L;

3.3.2 运动平台类型

```
enum Stage {XYR, XXY, XYY, XXYY, XXYY_LLRL};
```

在对位的应用中，大部分的运动平台为 XYR（R 代表旋转平台）或者采用 XXY 和 XXYY 平台，为了方便大家在使用相关函数的时候，我们在对位函数 `SY_MD_Free_Align` 和物理量转换为脉冲量的函数 `SY_MD_Physic_To_Pulse` 中引入了平台的类型，通过选定使用的平台类型后，函数会计算对应各轴的脉冲量或物理量（详见结构体 `PULSE` 和 `AlignData` 的说明）。

3.3.3 脉冲结构变量

```
typedef struct PULSE  
{  
    //XYR  
    float X;  
    float Y;  
    float R;  
  
    //XXY & XYY & XXYY  
    float X1;  
    float X2;  
    float Y1;  
    float Y2;  
  
    float X1mm;  
    float X2mm;  
    float Y1mm;  
    float Y2mm;  
}PULSE;
```

【1】XYR 平台

X: X 轴的位移量，单位为脉冲；Y: Y 轴的位移量，单位为脉冲；R: 旋转轴的位移量，单位为脉冲。

【2】XXY 或 XYY 平台

X1: X1 轴的位移量，单位为脉冲；X2 为 X2 轴的位移量，单位为脉冲；Y1: Y 轴的位移量，单位为脉冲。

X1: X1 轴的位移量，单位为脉冲；Y1 为 Y1 轴的位移量，单位为脉冲；Y2: Y2 轴的位移量，单位为脉冲。

【3】XXYY 平台

X1: X1 轴的位移量，单位为脉冲；X2 为 X2 轴的位移量，单位为脉冲；Y1: Y1 轴的位移量，单位为脉冲；Y2: Y2 轴的位移量，单位为脉冲。

由于对位应用中所用到的对位平台有 XYR 和 XXY 以及 XXYY 甚至一些较为复杂的对位平台，为了方便使用，我们设计 `SY_MD_Physic_To_Pulse` 函数用于将物理量转换为相应的脉冲量，方便用户直接给运动控制卡或者其他运动控制器直接下达脉冲位置信息即可完成相应的移动量。

3.3.4 五点标定坐标结构变量

MicroDistPro 提供此结构来配合外部抓取 Mark 点，完成自标定，确定平台和相机之间的关系。形式如下：

```
typedef struct Point10dF
{
    float ORGX;
    float ORGY;
    float PX_X;
    float PX_Y;
    float PY_X;
    float PY_Y;
    float CWX;
    float CWY;
    float CCWX;
    float CCWY;
}Point10dF;
```

参数说明：

ORGX 和 ORGY：相机抓取原点坐标的像素坐标。

PX_X 和 PX_Y：控制运动轴从原点向 X 方向移动后，抓取的像素坐标。

PY_X 和 PY_Y：控制运动轴从原点向 Y 方向移动后，抓取的像素坐标。

CWX 和 CWY：设备正方向旋转后，抓取的像素坐标。

CCWX 和 CCWY：设备负方向旋转后，抓取的像素坐标。

此结构体一般配合 `SY_MD_Free_Calibrate(float m_XYdist, float m_Angle, Point10dF MPoint)` 来使用。

3.3.5 对位结果结构变量

MicroDistPro 提供此结构来获得运动偏移量的信息，即 X、Y 和旋转轴的偏移量，单位以脉冲或物理量单位给出。形式如下：

```
typedef struct AlignData
{
    float xmm;           //mm
    float ymm;           //mm
    float angle;         //deg

    double org_width;    //
    double org_height;   //

    double samp_width;
    double samp_height;
}AlignData;
```

参数说明：

xmm: X 方向偏移量，单位为“mm”。

ymm: Y 方向偏移量，单位为“mm”。

angle: 角度偏移量，单位度。

org_width: 保留参数，不需要使用。

org_height: 保留参数，不需要使用。

samp_width: 保留参数，不需要使用。

samp_height: 保留参数，不需要使用。

3.3.6 标定信息结构变量

MicroDistPro 通过执行标定函数后，统一相机和平台的坐标系，使用 CaliInfos 此结构来获得当前相机的分辨率。形式如下：

```
typedef struct CamInfo
{
    int    c_width;      //Image width
    int    c_height;     //Image height
    float  f_width;      //FOV width (mm)
    float  f_height;     //FOV height (mm)
    float  xp;           //um/pixel in Camera X axis (mm/pixel)
    float  yp;           //un/pixel in Camera Y axis (mm/pixel)
}CamInfo;
```

参数说明：

`c_width` 和 `c_height`：图像的长度和宽度【单位为像素】。

`f_width` 和 `f_height`：视野大小【单位为“mm”】。

`xp` 和 `yp`：相机水平方向和垂直方向的像素精度【单位 um/Pixel】。

3.3.7 标定直线结构变量与直线标定结果结构变量

MicroDistPro 执行直线标定函数时，使用 `CalibrationLines` 此结构来记录 5 条用于标定的直线，并最终用 `LineCalibration` 结构体记录标定结果。形式如下：

```
struct CalibrationLines
{
    LineData OrgLine;//ORG
    LineData XPLine;//X+
    LineData YPLine;//Y+
    LineData RPLine;//R+
    LineData RMLine;//R-
};
```

```
struct LineCalibration
{
    double V[18];
};
```

参数说明：

`OrgLine`：标定起点处的直线信息。

`XPLine`：设备沿着 X 正方向移动后的直线信息。

`YPLine`：设备沿着 Y 正方向移动后的直线信息。

`RPLine`：设备正方向旋转后的直线信息。

`RMLine`：设备反方向旋转后的直线信息。

3.3.8 自标定结果结构变量

MicroDistPro 执行自标定函数时，使用 `Point5dVec` 此结构来记录标定的结果。形式如下：

```
typedef struct Point5dVec
{
    double V1;
    double V2;
    double V3;
    double V4;
    double V5;
};
```

```
}Point5dVec;
```

参数说明:

V1: 标定的旋转中心 X 坐标。

V2: 标定的旋转中心 Y 坐标。

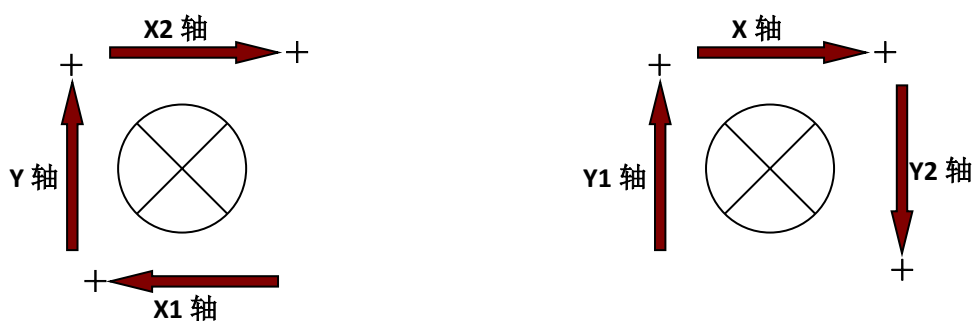
V3: 相机 X 轴与平台 X 轴之间的夹角 (弧度)。

V4: X 方向的像素分辨率。

V5: Y 方向的像素分辨率。

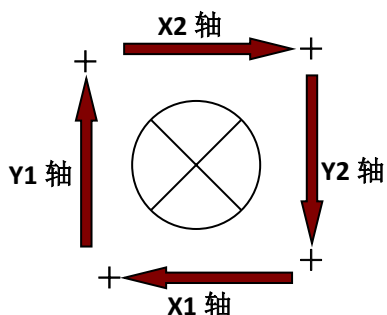
3.4 关于 XXY、XYY 与 XXYY 平台

UVW 平台的三轴结构与方向搭配多样, 为了方便客户使用 **MicroDistPro** 的函数及管理 UVW 平台, **MicroDistPro** 将 UVW 平台定义为 XXY 型平台, 且规定三轴的正方向为使得 XXY 平台顺时针方向旋转的方向, 如下图所示:



图中, 列举了 XXY、XYY 平台的结构, 三轴的同方向运动使平台能作顺时针或逆时针旋转运动。

同样地, 对于四轴的 XXYY 平台, 我们规定四轴的正方向为使得平台顺时针方向旋转的方向, 如下图所示:



四轴的同方向运动使平台能作顺时针或逆时针旋转运动。