SigmaStar
CALIB lib API User Guide

Version 1.1

REVISION HISTORY

Revision No.	Description	Date
1.0	● Initial release	04/20/2023
1.1	● Update MI_CALIB_Conf_t structure	10/18/2023

TABLE OF CONTENT

1. OVERVIEW

1.1.Algorithm Description

CALIB API includes calibration, Auto Fine-Tune, among other modules.

2. API REFERENCE

2.1.Function Module API

API Name	Function
MI_CALIB_Init	The memory required for the initialization of CALIB lib.
MI_CALIB_Deinit	The memory required for the de-initialization of CALIB lib.
MI_CALIB_Calibration	Enable model calibration and production calibration.
MI_CALIB_FineTune	Enable Auto Finetune.

2.2.MI_CALIB_Init

- **Function**

  The memory required for the initialization of CALIB lib.

- **Syntax**

  MI_CALIB_RetCode_e MI_CALIB_Init(MI_CALIB_Handle_t *pAlgHandle, MI_CALIB_InitPara_t *pstInitPara);

- **Parameter**

Parameter Name	Description	Input/Output
pAlgHandle	CALIB lib handle	Output
pstInitPara	Pointer to the initialization structure of CALIB lib. Currently not in use.	Input

- **Return Value**

Return Value	Result
E_MI_CALIB_SUCCESS	Success.
E_MI_CALIB_INVALID_PARA	Failed. Invalid parameter.
E_MI_CALIB_ERROR	Failed. Other errors.

➢ **Dependency**

- Header File: mi_calib.h
- Library File: Calib.lib

※ **Note**

N/A.

➢ **Example**

```
MI_CALIB_Handle_t algHandle = NULL;
MI_CALIB_Init(&algHandle, NULL);
```

2.3.MI_CALIB_Deinit

➢ **Function**

The memory required for the de-initialization of CALIB lib.

➢ **Syntax**

MI_CALIB_RetCode_e MI_CALIB_Deinit(MI_CALIB_Handle_t algHandle);

➢ **Parameter**

Parameter Name	Description	Input/Output
algHandle	CALIB lib handle	Input

➢ **Return Value**

Return Value	Result
E_MI_CALIB_SUCCESS	Success.
E_MI_CALIB_INVALID_PARA	Failed. Invalid parameter.
E_MI_CALIB_ERROR	Failed. Other errors.

➢ **Dependency**

- Header File: mi_calib.h
- Library File: Calib.lib

※ **Note**

N/A.

➢ **Example**

N/A.

2.4.MI_CALIB_Calibration

➢ **Function**

Enable model calibration and production calibration.

➢ **Syntax**

MI_CALIB_RetCode_e MI_CALIB_Calibration(MI_CALIB_Handle_t algHandle, MI_CALIB_Conf_t *pMI_Conf, MI_CALIB_BufBlock_t *pstInJsonBuf, MI_CALIB_BufBlock_t *pstOutJsonBuf, MI_CALIB_BufBlock_t *pcalibErrBuf);

➢ **Parameter**

Parameter Name	Description	Input/Output
algHandle	CALIB lib handle	Input
pMI_Conf	The pointer to CALIB lib calibration structure.	Output
pstInJsonBuf	The input configuration of json. Please refer to Calib_in.json.	Input
pstOutJsonBuf	The output configuration of json. Please refer to Calib_out.json.	Output
pcalibErrBuf	The error of calibration output. Please refer to reprojection_err.txt.	Output

➢ **Return Value**

Return Value	Result
E_MI_CALIB_SUCCESS	Success.
E_MI_CALIB_INVALID_PARA	Failed. Invalid parameter.
E_MI_CALIB_ERROR	Failed. Other errors.

➢ **Dependency**

- Header File: mi_calib.h

- Library File: Calib.lib

※ **Note**

- N/A.

➢ **Example**

```
MI_CALIB_RetCode_e MI_CALIB_ReadJsFile(const char *jsFile, MI_CALIB_BufBlock_t* pstJsonBuf)
{
    struct stat fileInfo;
    FILE * fp = NULL;
    unsigned int readBytes = 0;

    int result = stat(jsFile, &fileInfo);
    if (result < 0)
    {
        printf("stat file error!\n");
    }

    pstJsonBuf->pBufAddr = (void *)MI_CALIB_Malloc(fileInfo.st_size);
    fp = fopen(jsFile, "r");

    readBytes = fread(pstJsonBuf->pBufAddr, 1, fileInfo.st_size, fp);
    printf("readBytes:%d\n", readBytes);
    printf("st_size:%d\n", fileInfo.st_size);
    if (readBytes != fileInfo.st_size)
    {
        printf("read json file faild!\n");
    }
    pstJsonBuf->u32BufSz = readBytes;

    return E_MI_CALIB_SUCCESS;
}

MI_CALIB_RetCode_e MI_CALIB_WriteJsFile(const char *jsFile, MI_CALIB_BufBlock_t *pJsonBuf)
{
    char *pBuf = (char *)pJsonBuf->pBufAddr;
    FILE *fp = fopen(jsFile, "w");
    fwrite(pBuf, 1, strlen(pBuf), fp);
    fclose(fp);

    return E_MI_CALIB_SUCCESS;
}

MI_CALIB_RetCode_e MI_CALIB_FreeBlockBuf(MI_CALIB_BufBlock_t* pstJsonBuf)
{
    if (pstJsonBuf->pBufAddr != NULL)
    {
        mem_free(pstJsonBuf->pBufAddr);
        pstJsonBuf->u32BufSz = 0;
    }

    return E_MI_CALIB_SUCCESS;
}

void Calib_DEMO(const char *JSpath, const char *calibOutPath, const char *projectionErrPath)
{
    MI_CALIB_Handle_t algHandle = NULL;
```

```
    MI_CALIB_BufBlock_t stInJsonBuf, stOutJsonBuf, calibErrBuf;
    MI_CALIB_Conf_t mi_calib_conf;

    MI_CALIB_Init(&algHandle, NULL);
    MI_CALIB_ReadJsFile(JSpath, &stInJsonBuf);
    MI_CALIB_Calibration(algHandle, &mi_calib_conf, &stInJsonBuf, &stOutJsonBuf, &calibErrBuf);
    MI_CALIB_Deinit(algHandle);

    MI_CALIB_WriteJsFile(calibOutPath, &stOutJsonBuf);
    MI_CALIB_WriteJsFile(projectionErrPath, &calibErrBuf);

    MI_CALIB_FreeJsFile(&stInJsonBuf);
    MI_CALIB_FreeJsFile(&stOutJsonBuf);
    MI_CALIB_FreeJsFile(&calibErrBuf);
}

int main(int argc, char **argv)
{
     Calib_DEMO("Calib_in.json", "Calib_out.json", "reprojection_err.txt");
}
```

2.5.MI_CALIB_FineTune

- **Function**

  Enable Auto Finetune function.

- **Syntax**

  MI_CALIB_RetCode_e MI_CALIB_FineTune(MI_CALIB_Handle_t algHandle,
  const char *confPath,
  MI_CALIB_BufBlock_t *pstInJsonBuf,
  MI_CALIB_BufBlock_t *pstCalibJsonBuf,
  int *stateflag);

- **Parameter**

Parameter Name	Description	Input/Output
algHandle	CALIB lib handle	Input
confPath	Configuration file of Finetune. Please refer to sysConfig.ini.	Input
pstInJsonBuf	The input configuration of json. Please refer to Finetune_in.json.	Input
pstOutJsonBuf	The input configuration of Calib_out.json. Please refer to Calib_out.json.	Input/Output
stateflag	Finetune state flag	Output

➢ **Return Value**

Return Value	Result
E_MI_CALIB_SUCCESS	Success.
E_MI_CALIB_INVALID_PARA	Failed. Invalid parameter.
E_MI_CALIB_ERROR	Failed. Other errors.

➢ **Dependency**

- Header File: mi_calib.h
- Library File: Calib.lib

※ **Note**

- You must call MI_CALIB_Calibration to generate the result of model calibration before calling MI_CALIB_FineTune.

➢ **Example**

```
void Finetune_DEMO(const char *JSpath, const char *calibOutPath, const char *sysConfig, const char
*finetuneOutPath)
{
    MI_CALIB_Handle_t algHandle = NULL;
    MI_CALIB_BufBlock_t stInJsonBuf, stOutJsonBuf;
    int ftstateflag[10];

    MI_CALIB_Init(&algHandle, NULL);
    MI_CALIB_ReadJsFile(JSpath, &stInJsonBuf);
    MI_CALIB_ReadJsFile(calibOutPath, &stOutJsonBuf);

    MI_CALIB_FineTune(algHandle, sysConfig, &stInJsonBuf, &stOutJsonBuf, ftstateflag);

    MI_CALIB_WriteJsFile(finetuneOutPath, &stOutJsonBuf);
    MI_CALIB_Deinit(algHandle);
}

int main(int argc, char **argv)
{
     Calib_DEMO("Finetune_in.json", "Calib_out.json", "projectionErr.txt");
     Finetune_DEMO("Finetune_in.json", "Calib_out.json", "sysConfig.ini", "Finetune_out.json");
}
```

3. CALIB DATA TYPE

3.1.Definition of Data Type of CALIB Module

Data Type	Definition
MI_CALIB_Conf_t	The structure of CALIB lib calibration configuration.
MI_CALIB_BufBlock_t	The structure of buffer block used by CALIB lib.
MI_CALIB_BufType_e	The type of CALIB lib buffer block.
MI_CALIB_RetCode_e	The type of CALIB lib return value.
MI_CALIB_Handle_t	The type of CALIB lib handle.
MI_CALIB_PhyAddr_t	The type of physical address of CALIB lib buffer block.

3.2.MI_CALIB_Conf_t

- **Description**

  Define the structure of CALIB lib calibration configuration.

- **Definition**

```
typedef struct MI_CALIB_Conf_t
{
    int camNum;
    int camPairNum;
    int camLensType;
    int imgWidth;
    int imgHeight;
    int inputType;
    int intrBoardWidth;
    int intrBoardHeight;
    int intrBoardSize;
    int extrBoardWidth;
    int extrBoardHeight;
    int extrBoardSize;

    int blendType;
```

```
int mapProjType;
int mapCropType;
int checkCalibMode;
int fastCalibMode;
int productCalibMode;
int productTuneMode;
double reserveAlpha;
int seamFunnel;
int waveCalib;
int fixDist;
int dumpMode;

// DPU
int edge_width;
int edge_height;
int tile_width;
int tile_StepX;
int min_d;
int num_d;
int en_dpu_crop;
char k_coef[200];

// map-gen
int orgWidth;
int orgHeight;
int scaleWidth;
int scaleHeight;
int in_seq_mode;
int out_seq_mode;
int src_mode;
int hw_mode;
int seprate_bin_mode;
char camRotation[200];
int gridSize;
int preCropType;

char ptguiPath[200];
```

```
char imgPath[200];
char paraPath[200];
char outputPath[200];
char moduleName[200];
char imgFormat[200];
char imgPrefix[200];
char imgDepthList[200];
char camIDList[200];
char camPairIDList[2][200];
char camCenterID[200];
char rotateAngleList[200];
char blendWidthPercentList[200];

char camOffsetX[200];
char camOffsetY[200];
char imgEdgeTop[200];
char imgEdgeBottom[200];
char imgEdgeLeft[200];
char imgEdgeRight[200];
char calibOutPath[200];
char configPath[200];

int VerticalNum;
int updateBinMode;
int autoCalcMode;
int FOVX;
int FOVY;
int prjCenterX;
int prjCenterY;
int yaw;
int pitch;
int roll;
char mask_en;
char connect_type;
char stitch_mode;
int debug_mapgen;
```

```
    int genMapType;
    int gridsize;
    int rotationtype;
    int crop_En;
    int correctAlpha;
    int correctBeta;
    int cropwidth;
    int cropheight;
    int offx;
    int offy;
} MI_CALIB_Conf;
```

- **Calib Members**

Member Name	Description
camNum	The number of camera.
camPairNum	The pair number of camera.
camLensType	The lens type of camera. 0: wide-angle lens; 1: super-wide-angle lens.
imgWidth	The width of image.
imgHeight	The height of image.
inputType	The calibration type. 0: model calibration; 1: production line calibration; 2: PTGui; 3: Hugin.
intrBoardWidth	The number of horizontal corner points on the checkerboard during intrinsic parameter calibration.
intrBoardHeight	The number of vertical corner points on the checkerboard during intrinsic parameter calibration.
intrBoardSize	The size of checkerboard square during intrinsic parameter calibration (in the unit of mm).
extrBoardWidth	The number of horizontal corner points on the checkerboard during extrinsic parameter calibration.
extrBoardHeight	The number of vertical corner points on the checkerboard during extrinsic parameter calibration.
extrBoardSize	The size of checkerboard square during extrinsic parameter calibration (in the unit of mm).
ptguiPath	Path to ptGui/Hugin parameter.
imgPath	Path to calibration image.

Member Name	Description
outputPath	Path to output image.
imgPrefix	The prefix of image for filtering purpose.
camIDList	The list of camera id.
camPairIDList	The list of camera pair id.
camCenterID	The ID of center camera.
rotateAngleList	The list of camera rotation angle.
calibOutPath	Path to calibration output.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration

3.3.MI_CALIB_BufBlock_t

➢ **Description**

Define the structure of buffer block used by CALIB lib.

➢ **Definition**

```
typedef struct
{
    void *pBufAddr;
    MI_CALIB_PhyAddr_t phyAddr;
    uint32_t u32BufSz;
    MI_CALIB_BufType_e eBufType;
} MI_CALIB_BufBlock_t;
```

➢ **Member**

Member Name	Description
pBufAddr	The virtual address of memory block.
phyAddr	The physical address of memory block address. This parameter is invalid in the offline library.
u32BufSz	The size of memory block.
eBufType	The type of memory block. This parameter is invalid in the offline library.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration, MI_CALIB_FineTune

3.4.MI_CALIB_BufType_e

➢ **Description**

Define the type of CALIB lib buffer block.

➢ **Definition**

```
typedef enum
{
    E_MI_CALIB_BUFTYPE_NONCONTINUED = 0,
    E_MI_CALIB_BUFTYPE_CONTINUED = 1
} MI_CALIB_BufType_e;
```

➢ **Member**

Member Name	Description
E_MI_CALIB_BUFTYPE_NONCONTINUED	Non-contiguous memory. The offline algorithm library only supports non-contiguous memory.
E_MI_CALIB_BUFTYPE_CONTINUED	Contiguous memory.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration, MI_CALIB_FineTune

3.5.MI_CALIB_RetCode_e

➢ **Description**

Define the type of CALIB lib return value.

➢ **Definition**

```
typedef enum
{
    E_MI_CALIB_SUCCESS = 0,
    E_MI_CALIB_INVALID_PARA = 1,
    E_MI_CALIB_ERROR = 2
} MI_CALIB_RetCode_e;
```

➢ **Member**

Member Name	Description
E_MI_CALIB_SUCCESS	Success.
E_MI_CALIB_INVALID_PARA	Failed. Invalid parameter.
E_MI_CALIB_ERROR	Failed. Other errors.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Init, MI_CALIB_Deinit, MI_CALIB_Calibration, MI_CALIB_FineTune

3.6.MI_CALIB_Handle_t

➢ **Description**

Define the type of CALIB lib handle.

➢ **Definition**

typedef void *MI_CALIB_Handle_t;

➢ **Member**

Member Name	Description

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Init, MI_CALIB_Deinit, MI_CALIB_Calibration, MI_CALIB_FineTune

3.7.MI_CALIB_PhyAddr_t

➢ **Description**

Define the type of physical address of CALIB lib buffer block.

➢ **Definition**

typedef unsigned long long int MI_CALIB_PhyAddr_t;

➢ **Member**

Member Name	Description

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration, MI_CALIB_FineTune

4. CALIB CONFIGURATION FILE

4.1.CALIB Module Configuration File

Data Type	Definition
Calib_in.json	Json configuration file of calibration input.
Calib_out.json	Json configuration file of calibration output.
reprojection_err.txt	Projection error file of calibration output.
Finetune_in.json	Json configuration file of Finetune input.
Finetune_out.json_AudioAecInit	Json configuration file of Finetune output.
sysConfig.ini	Parameter file of Finetune algorithm.

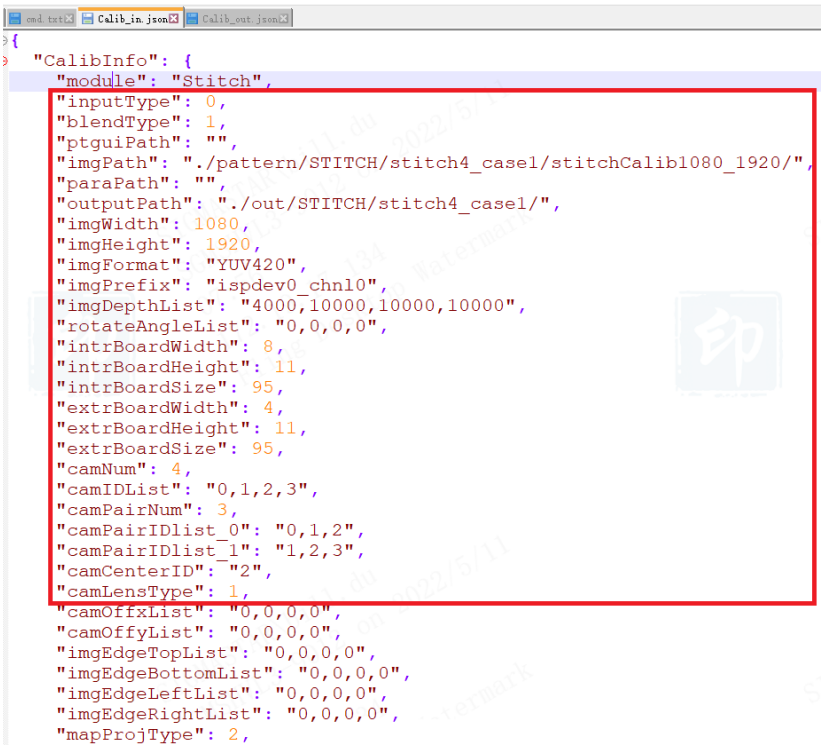
4.2.Calib_in.json

- **Description**

  Define the Json configuration file of calibration input.

- **Definition**

Calib_in.json



```
{
  "CalibInfo": {
    "module": "Stitch",
    "inputType": 0,
    "blendType": 1,
    "ptguiPath": "",
    "imgPath": "./pattern/STITCH/stitch4_case1/stitchCalib1080_1920/",
    "paraPath": "",
    "outputPath": "./out/STITCH/stitch4_case1/",
    "imgWidth": 1080,
    "imgHeight": 1920,
    "imgFormat": "YUV420",
    "imgPrefix": "ispdev0_chnl0",
    "imgDepthList": "4000,10000,10000,10000",
    "rotateAngleList": "0,0,0,0",
    "intrBoardWidth": 8,
    "intrBoardHeight": 11,
    "intrBoardSize": 95,
    "extrBoardWidth": 4,
    "extrBoardHeight": 11,
    "extrBoardSize": 95,
    "camNum": 4,
    "camIDList": "0,1,2,3",
    "camPairNum": 3,
    "camPairIDlist_0": "0,1,2",
    "camPairIDlist_1": "1,2,3",
    "camCenterID": "2",
    "camLensType": 1,
    "camOffxList": "0,0,0,0",
    "camOffyList": "0,0,0,0",
    "imgEdgeTopList": "0,0,0,0",
    "imgEdgeBottomList": "0,0,0,0",
    "imgEdgeLeftList": "0,0,0,0",
    "imgEdgeRightList": "0,0,0,0",
    "mapProjType": 2,
```

- **Calib Members**

Member Name	Description
camNum	The number of camera. For binocular camera, fill in 2; for trinocular camera, fill in 3.
camPairNum	The number of camera pair. For binocular camera, fill in 1; for trinocular camera, fill in 2.
camLensType	The type of camera lens. 0: wide-angle lens; 1: super-wide-angle lens.
imgWidth	The width of image, in the unit of pixel.
imgHeight	The height of image, in the unit of pixel.
imgFormat	The format of image, RGB/YUV420.
inputType	The type of calibration. 0: Model calibration; 1: production line calibration; 3: Auto-FineTune.
intrBoardWidth	The number of horizontal corner points on the checkerboard during intrinsic parameter calibration.
intrBoardHeight	The number of vertical corner points on the checkerboard during intrinsic parameter calibration.
intrBoardSize	The size of checkerboard square during intrinsic parameter calibration (in the unit of mm).

Member Name	Description
extrBoardWidth	The number of horizontal corner points on the checkerboard during extrinsic parameter calibration.
extrBoardHeight	The number of vertical corner points on the checkerboard during extrinsic parameter calibration.
extrBoardSize	The size of checkerboard square during extrinsic parameter calibration (in the unit of mm).
ptguiPath	Path to ptGui/Hugin parameter.
imgPath	Path to calibration image.
outputPath	Path to output result.
imgPrefix	The prefix of image for filtering purpose.
camIDList	The list of camera id.
camPairIDList	The list of camera pair id.
camCenterID	The ID of center camera.
rotateAngleList	The list of camera rotation angle.
module	Module name. Function: Stitch – stitching; NIR – Near Infrared fusion.
dumpMode	Whether to dump the corner detection image: 0-disable 1-enable

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration

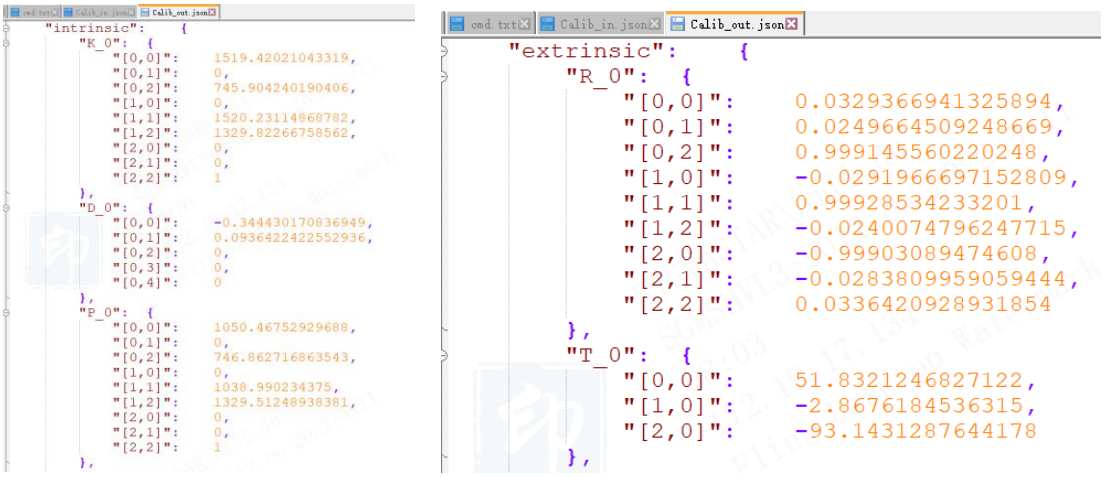
4.3.Calib_out.json

➢ **Description**

Define the Json configuration file of calibration output.

➢ **Definition**

Calib_out.json



➢ **Member**

Member Name	Description
intrinsic K_i D_i P_i	Intrinsic parameter K, D, and P of camera i.
extrinsic R_i T_i	Extrinsic parameter R and T of camera pair i.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration, MI_CALIB_FineTune

4.4.reprojection_err.txt

➢ **Description**

Define the projection error file of calibration output.

➢ **Definition**

reprojection_err.txt

cmd.txt | reprojection_err.txt

```
0.133096, 0.581686
0.118579, 0.487245
0.134781, 0.639467
0.113132, 0.481265
0.094774, 0.407415
0.091279, 0.370978
0.085691, 0.246767
```

➢ **Member**

Member Name	Description
The projection error of each camera	The average projection error of camera i The maximum projection error
The projection error of each camera pair	The average projection error of camera pair i The maximum projection error

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_Calibration, MI_CALIB_FineTune

4.5.Finetune_in.json

- **Description**

  Define the Json configuration file of Finetune input.

- **Definition**

finetune_in.json

```
{
  "finetune":    {
    "module":   "Stitch",
    "ftimgPath":  "./pattern/STITCH/stitch4_case3/finetune/pattern6/",
    "ftoutputPath":    "./pattern/STITCH/stitch4_case3/finetune/pattern6/out/",
    "ftimgFormat":    "YUV420",
    "ftimgPrefix":    "ispdev0_chnl0",
    "imgWidth": 1920,
    "imgHeight":    1080,
    "imgFormat":    "YUV420",
    "imgPrefix":    "ispdev0_chnl0",
    "imgDepthList": "5000,5000,5000,5000",
    "rotateAngleList": "270,270,270,270",
    "camNum":   4,
    "camIDList":    "0,1,2,3,",
    "camPairNum":   3,
    "camPairIDlist_0":  "0,1,2,",
    "camPairIDlist_1":  "1,2,3,",
    "camCenterID":  "2"
  },
```

- **Calib Members**

Member Name	Description
camNum	The number of camera.
camPairNum	The number of camera pair.
imgWidth	The width of image.
imgHeight	The height of image.
ftimgPath	The path to Finetune image.
ftimgFormat	The format of image, RGB/YUV420.
ftoutputPath	The path to Finetune output result.
ftimgPrefix	The prefix of image for filtering purpose.
camIDList	The list of camera id.
camPairIDList	The list of camera pair id.
camCenterID	The ID of center camera.
rotateAngleList	The list of camera rotation angle.
module	Module name. For stitch function, select “Stitch”.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_FineTune

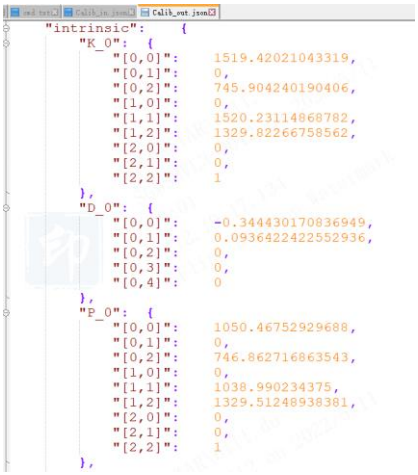
4.6.Finetune_out.json

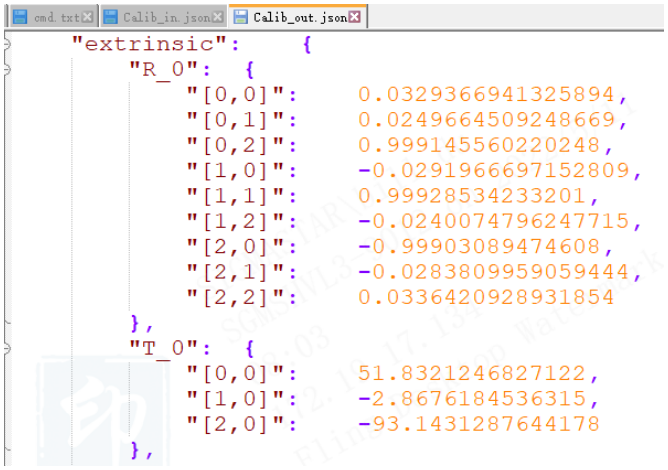
➢ **Description**

Define the Json configuration file of Finetune output.

➢ **Definition**

finetune_out.json





➢ **Member**

Member Name	Description
intrinsic K_i D_i P_i	Intrinsic parameter K, D, and P of camera i.
extrinsic R_i T_i	Extrinsic parameter R and T of camera pair i.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_FineTune

4.7.sysConfig.ini

➢ **Description**

Define the parameter file of Finetune algorithm.

➢ **Definition**


sysConfig.ini

➢ **Member**

Member Name	Description
intrinsic K_i D_i P_i	Intrinsic parameter K, D, and P of camera i.
extrinsic R_i T_i	Extrinsic parameter R and T of camera pair i.

※ **Note**

N/A.

➢ **Related Data Type and Interface**

MI_CALIB_FineTune