TYCamport3

3

Generated by Doxygen 1.8.17

1	Main Page	1
	1.1 Note	1
2	Class Index	3
	2.1 Class List	3
3	File Index	5
	3.1 File List	5
4	Class Documentation	7
	4.1 DepthEnhenceParameters Struct Reference	7
	4.1.1 Detailed Description	7
	4.2 DepthSpeckleFilterParameters Struct Reference	7
	4.2.1 Detailed Description	7
	4.3 gz_header_s Struct Reference	8
	4.3.1 Detailed Description	8
	4.4 gzFile_s Struct Reference	8
	4.4.1 Detailed Description	8
	4.5 pattern_bin_param Struct Reference	8
	4.5.1 Detailed Description	9
	4.6 pattern_gray_param Struct Reference	9
	4.6.1 Detailed Description	9
	4.7 pattern_sine_param Struct Reference	9
	4.7.1 Detailed Description	9
	4.8 TY_PHC_GROUP_ATTR::phc_group_attr Struct Reference	10
	4.8.1 Detailed Description	10
	4.9 TY_ACC_BIAS Struct Reference	10
	4.9.1 Detailed Description	10
	4.10 TY_ACC_MISALIGNMENT Struct Reference	10
	4.10.1 Detailed Description	11
	4.11 TY_ACC_SCALE Struct Reference	11
	4.11.1 Detailed Description	11
	4.12 TY_AEC_ROI_PARAM Struct Reference	12
	4.12.1 Detailed Description	12
	4.13 TY_BYTEARRAY_ATTR Struct Reference	12
	4.13.1 Detailed Description	12
	4.13.2 Member Data Documentation	12
	4.13.2.1 unit_size	13
	4.13.2.2 valid_size	13
	4.14 TY_CAMERA_CALIB_INFO Struct Reference	13
	4.14.1 Detailed Description	14
	4.15 TY_CAMERA_DISTORTION Struct Reference	14
	4.15.1 Detailed Description	14

4.16 TY_CAMERA_EXTRINSIC Struct Reference	14
4.16.1 Detailed Description	14
4.17 TY_CAMERA_INTRINSIC Struct Reference	15
4.17.1 Detailed Description	15
4.18 TY_CAMERA_ROTATION Struct Reference	16
4.18.1 Detailed Description	16
4.19 TY_CAMERA_STATISTICS Struct Reference	16
4.19.1 Detailed Description	16
4.20 TY_CAMERA_TO_IMU Struct Reference	17
4.20.1 Detailed Description	17
4.21 TY_DEVICE_BASE_INFO Struct Reference	17
4.21.1 Detailed Description	18
4.22 TY_DEVICE_NET_INFO Struct Reference	18
4.22.1 Detailed Description	19
4.23 TY_DEVICE_USB_INFO Struct Reference	19
4.23.1 Detailed Description	19
4.24 TY_DI_WORKMODE Struct Reference	19
4.24.1 Detailed Description	19
4.25 TY_DO_WORKMODE Struct Reference	20
4.25.1 Detailed Description	20
4.26 TY_ENUM_ENTRY Struct Reference	20
4.26.1 Detailed Description	20
4.27 TY_EVENT_INFO Struct Reference	21
4.27.1 Detailed Description	21
4.28 TY_FEATURE_INFO Struct Reference	21
4.28.1 Detailed Description	21
4.29 TY_FLOAT_RANGE Struct Reference	22
4.29.1 Detailed Description	22
4.30 TY_FRAME_DATA Struct Reference	22
4.30.1 Detailed Description	23
4.31 TY_GYRO_BIAS Struct Reference	23
4.31.1 Detailed Description	23
4.32 TY_GYRO_MISALIGNMENT Struct Reference	23
4.32.1 Detailed Description	24
4.33 TY_GYRO_SCALE Struct Reference	24
4.33.1 Detailed Description	24
4.34 TY_IMAGE_DATA Struct Reference	25
4.34.1 Detailed Description	25
4.35 TY_IMU_DATA Struct Reference	25
4.35.1 Detailed Description	26
4.36 TY_INT_RANGE Struct Reference	26
4.36.1 Detailed Description	26

4.37 TY_INTERFACE_INFO Struct Reference	 26
4.37.1 Detailed Description	 27
4.38 TY_LASER_PARAM Struct Reference	 27
4.38.1 Detailed Description	 27
4.39 TY_LASER_PATTERN_PARAM Struct Reference	 27
4.39.1 Detailed Description	 28
4.40 TY_PHC_GROUP_ATTR Struct Reference	 28
4.40.1 Detailed Description	 29
4.41 TY_PIXEL_COLOR_DESC Struct Reference	 29
4.41.1 Detailed Description	 29
4.42 TY_PIXEL_DESC Struct Reference	 29
4.42.1 Detailed Description	 29
4.43 TY_TEMP_DATA Struct Reference	 30
4.43.1 Detailed Description	 30
4.44 TY_TOF_FREQ Struct Reference	 30
4.44.1 Detailed Description	 30
4.45 TY_TRIGGER_PARAM Struct Reference	 30
4.45.1 Detailed Description	 30
4.46 TY_TRIGGER_PARAM_EX Struct Reference	 31
4.46.1 Detailed Description	 31
4.47 TY_TRIGGER_TIMER_LIST Struct Reference	 31
4.47.1 Detailed Description	 31
4.48 TY_TRIGGER_TIMER_PERIOD Struct Reference	 32
4.48.1 Detailed Description	 32
4.49 TY_VECT_3F Struct Reference	 32
4.49.1 Detailed Description	 32
4.50 TY_VERSION_INFO Struct Reference	 32
4.50.1 Detailed Description	 33
4.51 TYEnumEntry Struct Reference	 33
4.51.1 Detailed Description	 33
4.52 z_stream_s Struct Reference	 33
4.52.1 Detailed Description	 33
5 File Decumentation	25
5 File Documentation	35
5.1 TYApi.h File Reference	35
5.1.1 Detailed Description	39
5.1.2 Function Documentation	39
5.1.2.1 TYAppendLogToFile()	40
5.1.2.2 TYAppendLogToServer()	40
5.1.2.3 TYClearBufferQueue()	41
5.1.2.4 TYCloseDevice()	41
5.1.2.5 TYCloseInterface()	 42

5.1.2.6 TYDeinitLib()
5.1.2.7 TYDisableComponents()
5.1.2.8 TYEnableComponents()
5.1.2.9 TYEnqueueBuffer()
5.1.2.10 TYErrorString()
5.1.2.11 TYFetchFrame()
5.1.2.12 TYForceDeviceIP()
5.1.2.13 TYGetBool()
5.1.2.14 TYGetByteArray()
5.1.2.15 TYGetByteArrayAttr()
5.1.2.16 TYGetByteArraySize()
5.1.2.17 TYGetComponentIDs()
5.1.2.18 TYGetDeviceFeatureInfo()
5.1.2.19 TYGetDeviceFeatureNumber()
5.1.2.20 TYGetDeviceInfo()
5.1.2.21 TYGetDeviceInterface()
5.1.2.22 TYGetDeviceList()
5.1.2.23 TYGetDeviceNumber()
5.1.2.24 TYGetDeviceXML()
5.1.2.25 TYGetDeviceXMLSize()
5.1.2.26 TYGetEnabledComponents()
5.1.2.27 TYGetEnum()
5.1.2.28 TYGetEnumEntryCount()
5.1.2.29 TYGetEnumEntryInfo()
5.1.2.30 TYGetFeatureInfo()
5.1.2.31 TYGetFloat()
5.1.2.32 TYGetFloatRange()
5.1.2.33 TYGetFrameBufferSize()
5.1.2.34 TYGetInt()
5.1.2.35 TYGetInterfaceList()
5.1.2.36 TYGetInterfaceNumber()
5.1.2.37 TYGetIntRange()
5.1.2.38 TYGetString()
5.1.2.39 TYGetStringLength()
5.1.2.40 TYGetStruct()
5.1.2.41 TYHasDevice()
5.1.2.42 TYHasFeature()
5.1.2.43 TYHasInterface()
5.1.2.44 TYLibVersion()
5.1.2.45 TYOpenDevice()
5.1.2.46 TYOpenDeviceWithIP()
5.1.2.47 TYOpenInterface()

5.1.2.48 TYRegisterEventCallback()	 87
5.1.2.49 TYRegisterImuCallback()	 87
5.1.2.50 TYRemoveLogFile()	 88
5.1.2.51 TYRemoveLogServer()	 88
5.1.2.52 TYSendSoftTrigger()	 89
5.1.2.53 TYSetBool()	 90
5.1.2.54 TYSetByteArray()	 91
5.1.2.55 TYSetEnum()	 93
5.1.2.56 TYSetFloat()	 94
5.1.2.57 TYSetInt()	 96
5.1.2.58 TYSetLogLevel()	 97
5.1.2.59 TYSetLogPrefix()	 97
5.1.2.60 TYSetString()	 98
5.1.2.61 TYSetStruct()	 99
5.1.2.62 TYStartCapture()	 101
5.1.2.63 TYStopCapture()	 102
5.1.2.64 TYUpdateAllDeviceList()	 103
5.1.2.65 TYUpdateDeviceList()	 103
5.1.2.66 TYUpdateInterfaceList()	 104
5.2 TYCoordinateMapper.h File Reference	 104
5.2.1 Detailed Description	 106
5.2.2 Macro Definition Documentation	 106
5.2.2.1 TYMAP_CHECKRET	 107
5.2.3 Function Documentation	 107
5.2.3.1 TYDepthImageFillEmptyRegion()	 107
5.2.3.2 TYInvertExtrinsic()	 107
5.2.3.3 TYMapDepthImageToPoint3d()	 108
5.2.3.4 TYMapDepthToPoint3d()	 108
5.2.3.5 TYMapPoint3dToDepth()	 109
5.2.3.6 TYMapPoint3dToDepthImage()	 109
5.2.3.7 TYMapPoint3dToPoint3d()	 110
5.3 TYDefs.h File Reference	 110
5.3.1 Detailed Description	 120
5.3.2 Macro Definition Documentation	 121
5.3.2.1 TY_DECLARE_IMAGE_MODE1	 121
5.3.3 Typedef Documentation	 121
5.3.3.1 TY_ACC_BIAS	 121
5.3.3.2 TY_ACC_MISALIGNMENT	 121
5.3.3.3 TY_ACC_SCALE	 122
5.3.3.4 TY_ACCESS_MODE_LIST	 122
5.3.3.5 TY_BYTEARRAY_ATTR	 122
5.3.3.6 TY_CAMERA_CALIB_INFO	 123

5.3.3.7 TY_CAMERA_DISTORTION	123
5.3.3.8 TY_CAMERA_EXTRINSIC	123
5.3.3.9 TY_CAMERA_INTRINSIC	123
5.3.3.10 TY_CAMERA_ROTATION	124
5.3.3.11 TY_CAMERA_TO_IMU	124
5.3.3.12 TY_COMPONENT_ID	124
5.3.3.13 TY_DEVICE_BASE_INFO	125
5.3.3.14 TY_DEVICE_COMPONENT_LIST	125
5.3.3.15 TY_ENUM_ENTRY	125
5.3.3.16 TY_FEATURE_ID	126
5.3.3.17 TY_FLOAT_RANGE	126
5.3.3.18 TY_GYRO_BIAS	126
5.3.3.19 TY_GYRO_MISALIGNMENT	126
5.3.3.20 TY_GYRO_SCALE	127
5.3.3.21 TY_INTERFACE_INFO	127
5.3.3.22 TY_INTERFACE_TYPE_LIST	127
5.3.3.23 TY_PIXEL_BITS_LIST	127
5.3.3.24 TY_TRIGGER_MODE_LIST	128
5.3.4 Enumeration Type Documentation	128
5.3.4.1 TY_ACCESS_MODE_LIST	128
5.3.4.2 TY_DEVICE_COMPONENT_LIST	128
5.3.4.3 TY_FEATURE_ID_LIST	129
5.3.4.4 TY_INTERFACE_TYPE_LIST	132
5.3.4.5 TY_PIXEL_BITS_LIST	132
5.3.4.6 TY_PIXEL_FORMAT_LIST	132
5.3.4.7 TY_RESOLUTION_MODE_LIST	133
5.3.4.8 TY_TRIGGER_MODE_LIST	134
5.4 TYImageProc.h File Reference	134
5.4.1 Detailed Description	136
5.4.2 Function Documentation	136
5.4.2.1 TYDepthEnhenceFilter()	136
5.4.2.2 TYDepthSpeckleFilter()	136
5.4.2.3 TYImageProcesAcceEnable()	137
5.4.2.4 TYUndistortImage()	137
Index	139

Chapter 1

Main Page

Copyright(C)2016-2023 Percipio All Rights Reserved

1.1 Note

Depth camera, called "device", consists of several components. Each component is a hardware module or virtual module, such as RGB sensor, depth sensor. Each component has its own features, such as image width, exposure time, etc..

NOTE: The component TY_COMPONENT_DEVICE is a virtual component that contains all features related to the whole device, such as trigger mode, device IP.

Each frame consists of several images. Normally, all the images have identical timestamp, means they are captured at the same time.

2 Main Page

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

DepthEnhenceParameters
DepthSpeckleFilterParameters
gz_header_s
gzFile_s
pattern_bin_param
pattern_gray_param
pattern_sine_param
TY_PHC_GROUP_ATTR::phc_group_attr
TY_ACC_BIAS
TY_ACC_MISALIGNMENT
TY_ACC_SCALE
TY_AEC_ROI_PARAM
TY_BYTEARRAY_ATTR
Byte array data structure
TY_CAMERA_CALIB_INFO 13
TY_CAMERA_DISTORTION
TY_CAMERA_EXTRINSIC
TY_CAMERA_INTRINSIC
TY_CAMERA_ROTATION
TY_CAMERA_STATISTICS 16
TY_CAMERA_TO_IMU
TY_DEVICE_BASE_INFO
TY_DEVICE_NET_INFO
Device network information
TY_DEVICE_USB_INFO
TY_DI_WORKMODE
TY_DO_WORKMODE 20
TY_ENUM_ENTRY
TY_EVENT_INFO
TY_FEATURE_INFO
TY_FLOAT_RANGE
Float range data structure
TY_FRAME_DATA
TY_GYRO_BIAS
TY GYRO MISALIGNMENT

4 Class Index

TY_GYRO_SCALE	24
TY_IMAGE_DATA	25
TY_IMU_DATA	25
TY_INT_RANGE	26
TY_INTERFACE_INFO	26
TY_LASER_PARAM	27
TY_LASER_PATTERN_PARAM	27
TY_PHC_GROUP_ATTR	28
TY_PIXEL_COLOR_DESC	29
TY_PIXEL_DESC	29
TY_TEMP_DATA	30
TY_TOF_FREQ	30
TY_TRIGGER_PARAM	30
TY_TRIGGER_PARAM_EX	31
TY_TRIGGER_TIMER_LIST (31
TY_TRIGGER_TIMER_PERIOD	32
TY_VECT_3F	32
TY_VERSION_INFO	32
TYEnumEntry	33
7 stream s	ď

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

I YApı.h	
TYApi.h includes camera control and data receiving interface, which supports configuration for	
image resolution, frame rate, exposure	
time, gain, working mode,etc	35
TYCoordinateMapper.h	
Coordinate Conversion API	104
TYDefs.h	
TYDefs.h includes camera control and data receiving data definitions which supports configura-	
tion for image resolution, frame rate, exposure	
time, gain, working mode,etc	110
TYImageProc.h	134
TYParameter.h	??
TYVer.h	??
zconf.h	??
zlib.h	

6 File Index

Chapter 4

Class Documentation

4.1 DepthEnhenceParameters Struct Reference

Public Attributes

- float sigma_s
 filter param on space
- float sigma_r

filter param on range

· int outlier_win_sz

outlier filter windows ize

• float outlier_rate

4.1.1 Detailed Description

Definition at line 53 of file TYImageProc.h.

The documentation for this struct was generated from the following file:

• TYImageProc.h

4.2 DepthSpeckleFilterParameters Struct Reference

Public Attributes

- int max_speckle_size
- int max_speckle_diff

4.2.1 Detailed Description

Definition at line 33 of file TYImageProc.h.

The documentation for this struct was generated from the following file:

• TYImageProc.h

4.3 gz_header_s Struct Reference

Public Attributes

- int text
- uLong time
- int xflags
- int os
- Bytef * extra
- uInt extra_len
- uInt extra_max
- Bytef * name
- uInt name max
- Bytef * comment
- uInt comm_max
- int hcrc
- int done

4.3.1 Detailed Description

Definition at line 114 of file zlib.h.

The documentation for this struct was generated from the following file:

· zlib.h

4.4 gzFile_s Struct Reference

Public Attributes

- unsigned have
- unsigned char * next
- z_off64_t **pos**

4.4.1 Detailed Description

Definition at line 1837 of file zlib.h.

The documentation for this struct was generated from the following file:

· zlib.h

4.5 pattern_bin_param Struct Reference

Public Attributes

- uint32_t offset
- uint8_t data [512]

4.5.1 Detailed Description

Definition at line 1085 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.6 pattern_gray_param Struct Reference

Public Attributes

- uint32_t phase_num
- uint32_t param1
- uint32_t param2
- · uint32_t param3

4.6.1 Detailed Description

Definition at line 1077 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.7 pattern_sine_param Struct Reference

Public Attributes

- uint32_t phase_num
- · float period

4.7.1 Detailed Description

Definition at line 1071 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.8 TY_PHC_GROUP_ATTR::phc_group_attr Struct Reference

Public Attributes

- uint8_t type
- uint8_t amp_thresh
- uint16_t ch
- uint8_t chn_type
- uint8_t rsvd [27]

4.8.1 Detailed Description

Definition at line 1055 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.9 TY_ACC_BIAS Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3]

4.9.1 Detailed Description

a 3x3 matrix

BIASx	BIASy	BIASz

Definition at line 1135 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.10 TY_ACC_MISALIGNMENT Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3 *3]

4.10.1 Detailed Description

a 3x3 matrix |.|.|.|

	•	
1	-GAMAyz	GAMAzy
GAMAxz	1	-GAMAzx
-GAMAxy	GAMAyx	1

Definition at line 1147 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.11 TY_ACC_SCALE Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3 *3]

4.11.1 Detailed Description

a 3x3 matrix

•		
SCALEx	0	0
0	SCALEy	0
0	0	SCALEz

Definition at line 1158 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.12 TY AEC ROI PARAM Struct Reference

Public Attributes

- uint32_t x
- uint32 t y
- uint32_t w
- uint32_t **h**

4.12.1 Detailed Description

Definition at line 1035 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.13 TY_BYTEARRAY_ATTR Struct Reference

byte array data structure

```
#include <TYDefs.h>
```

Public Attributes

• int32 t size

Bytes array size in bytes.

- int32_t unit_size
- int32_t valid_size

4.13.1 Detailed Description

byte array data structure

See also

TYGetByteArray

Definition at line 903 of file TYDefs.h.

4.13.2 Member Data Documentation

4.13.2.1 unit_size

int32_t TY_BYTEARRAY_ATTR::unit_size

unit size in bytes for special parse

Definition at line 906 of file TYDefs.h.

4.13.2.2 valid_size

int32_t TY_BYTEARRAY_ATTR::valid_size

valid size in bytes in case has reserved member, Must be multiple of unit_size, mem_length = valid_size/unit_size

Definition at line 909 of file TYDefs.h.

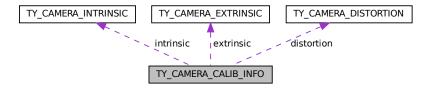
The documentation for this struct was generated from the following file:

• TYDefs.h

4.14 TY_CAMERA_CALIB_INFO Struct Reference

#include <TYDefs.h>

Collaboration diagram for TY_CAMERA_CALIB_INFO:



Public Attributes

- int32_t intrinsicWidth
- int32_t intrinsicHeight
- TY_CAMERA_INTRINSIC intrinsic
- TY_CAMERA_EXTRINSIC extrinsic
- TY_CAMERA_DISTORTION distortion

4.14.1 Detailed Description

camera 's cailbration data

See also

TYGetStruct

Definition at line 978 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.15 TY_CAMERA_DISTORTION Struct Reference

```
#include <TYDefs.h>
```

Public Attributes

• float data [12]

Definition is compatible with opencv3.0+ :k1,k2,p1,p2,k3,k4,k5,k6,s1,s2,s3,s4.

4.15.1 Detailed Description

camera distortion parameters

See also

```
TYGetStruct Usage:
TY_CAMERA_DISTORTION distortion;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_DISTORTION, &distortion, sizeof(distortion));
```

Definition at line 970 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.16 TY_CAMERA_EXTRINSIC Struct Reference

```
#include <TYDefs.h>
```

Public Attributes

float data [4 *4]

4.16.1 Detailed Description

a 4x4 matrix

r11	r12	r13	t1
r21	r22	r23	t2
r31	r32	r33	t3
0	0	0	1

See also

```
TYGetStruct Usage:
TY_CAMERA_EXTRINSIC extrinsic;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_EXTRINSIC, &extrinsic, sizeof(extrinsic));
```

Definition at line 958 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.17 TY_CAMERA_INTRINSIC Struct Reference

```
#include <TYDefs.h>
```

Public Attributes

• float data [3 *3]

4.17.1 Detailed Description

a 3x3 matrix



See also

```
TYGetStruct Usage:
TY_CAMERA_INTRINSIC intrinsic;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_INTRINSIC, &intrinsic, sizeof(intrinsic));
```

Definition at line 940 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.18 TY_CAMERA_ROTATION Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3 *3]

4.18.1 Detailed Description

a 3x3 matrix

•	•	•
r00	r01	r02
r10	r11	r12
r20	r21	r22

See also

```
TYGetStruct Usage:
TY_CAMERA_ROTATION rotation;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_ROTATION, &rotation, sizeof(rotation));
```

Definition at line 1316 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.19 TY_CAMERA_STATISTICS Struct Reference

Public Attributes

- uint64_t packetReceived
- uint64_t packetLost
- uint64_t imageOutputed
- uint64_t imageDropped
- uint8_t rsvd [1024]

4.19.1 Detailed Description

Definition at line 1109 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.20 TY_CAMERA_TO_IMU Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [4 *4]

4.20.1 Detailed Description

a 4x4 matrix

r11	r12	r13	t1
r21	r22	r23	t2
r31	r32	r33	t3
0	0	0	1

Definition at line 1201 of file TYDefs.h.

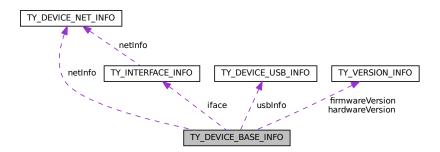
The documentation for this struct was generated from the following file:

• TYDefs.h

4.21 TY_DEVICE_BASE_INFO Struct Reference

#include <TYDefs.h>

Collaboration diagram for TY_DEVICE_BASE_INFO:



Public Attributes

4.21.1 Detailed Description

char buildHash [256]char configVersion [256]char reserved [256]

See also

TYGetDeviceList

Definition at line 841 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.22 TY_DEVICE_NET_INFO Struct Reference

```
device network information
```

```
#include <TYDefs.h>
```

Public Attributes

- char **mac** [32]
- char **ip** [32]
- char netmask [32]
- · char gateway [32]
- · char broadcast [32]
- · char tiversion [32]
- char reserved [64]

4.22.1 Detailed Description

device network information

Definition at line 812 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.23 TY DEVICE USB INFO Struct Reference

Public Attributes

- int bus
- int addr
- char reserved [248]

4.23.1 Detailed Description

Definition at line 823 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.24 TY_DI_WORKMODE Struct Reference

Public Attributes

- TY_E_DI_MODE mode
- TY_E_DI_INT_ACTION int_act
- uint32_t mode_supported
- uint32_t int_act_supported
- uint32_t status
- uint32_t reserved [3]

4.24.1 Detailed Description

Definition at line 1283 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.25 TY_DO_WORKMODE Struct Reference

Public Attributes

- TY_E_DO_MODE mode
- TY_E_VOLT_T volt
- uint32_t freq
- uint32 t duty
- uint32_t mode_supported
- uint32_t volt_supported
- uint32_t reserved [3]

4.25.1 Detailed Description

Definition at line 1260 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.26 TY_ENUM_ENTRY Struct Reference

#include <TYDefs.h>

Public Attributes

- char description [64]
- uint32_t value
- uint32_t reserved [3]

4.26.1 Detailed Description

enum feature entry information

See also

TYGetEnumEntryInfo

Definition at line 914 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.27 TY EVENT INFO Struct Reference

Public Attributes

- TY_EVENT eventId
- · char message [124]

4.27.1 Detailed Description

Definition at line 1254 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.28 TY_FEATURE_INFO Struct Reference

Public Attributes

· bool isValid

true if feature exists, false otherwise

TY ACCESS MODE accessMode

feature access privilege

bool writableAtRun

feature can be written while capturing

- char reserved0 [1]
- TY_COMPONENT_ID componentID

owner of this feature

• TY_FEATURE_ID featureID

feature unique id

• char name [32]

describe string

TY_COMPONENT_ID bindComponentID

component ID current feature bind to

TY_FEATURE_ID bindFeatureID

feature ID current feature bind to

- TY_VISIBILITY_TYPE visibility
- · char reserved [248]

4.28.1 Detailed Description

Definition at line 867 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.29 TY_FLOAT_RANGE Struct Reference

float range data structure

#include <TYDefs.h>

Public Attributes

- float min
- float max
- float inc

increaing step

• float reserved [1]

4.29.1 Detailed Description

float range data structure

See also

TYGetFloatRange

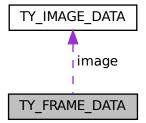
Definition at line 893 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.30 TY_FRAME_DATA Struct Reference

Collaboration diagram for TY_FRAME_DATA:



Public Attributes

void * userBuffer

Pointer to user enqueued buffer, user should enqueue this buffer in the end of callback.

· int32_t bufferSize

Size of userBuffer.

int32_t validCount

Number of valid data.

• int32_t reserved [6]

Reserved: reserved[0],laser_val;.

• TY_IMAGE_DATA image [10]

Buffer data, max to 10 images per frame, each buffer data could be an image or something else.

4.30.1 Detailed Description

Definition at line 1244 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.31 TY_GYRO_BIAS Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3]

4.31.1 Detailed Description

a 3x3 matrix

•	•	•
BIASx	BIASy	BIASz

Definition at line 1167 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.32 TY_GYRO_MISALIGNMENT Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3 *3]

4.32.1 Detailed Description

a 3x3 matrix

1	-ALPHAyz	ALPHAzy
0	1	-ALPHAzx
0	0	1

Definition at line 1178 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.33 TY_GYRO_SCALE Struct Reference

#include <TYDefs.h>

Public Attributes

• float data [3 *3]

4.33.1 Detailed Description

a 3x3 matrix

SCALEx	0	0
0	SCALEy	0
0	0	SCALEz

Definition at line 1189 of file TYDefs.h.

The documentation for this struct was generated from the following file:

4.34 TY IMAGE DATA Struct Reference

Public Attributes

uint64_t timestamp

Timestamp in microseconds.

• int32 t imageIndex

image index used in trigger mode

int32 t status

Status of this buffer.

TY_COMPONENT_ID componentID

Where current data come from.

• int32 t size

Buffer size.

void * buffer

Pointer to data buffer.

· int32_t width

Image width in pixels.

• int32_t height

Image height in pixels.

TYPixFmt pixelFormat

Pixel format, see TY_PIXEL_FORMAT_LIST.

int32_t reserved [9]

Reserved.

4.34.1 Detailed Description

Definition at line 1229 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.35 TY_IMU_DATA Struct Reference

Public Attributes

- uint64_t timestamp
- float acc x
- · float acc_y
- float acc_z
- float gyro_x
- · float gyro_y
- float gyro_z
- · float temperature
- float reserved [1]

4.35.1 Detailed Description

Definition at line 1118 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.36 TY_INT_RANGE Struct Reference

Public Attributes

- int32_t min
- int32_t max
- int32_t inc

increaing step

• int32_t reserved [1]

4.36.1 Detailed Description

Definition at line 883 of file TYDefs.h.

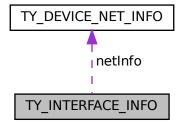
The documentation for this struct was generated from the following file:

• TYDefs.h

4.37 TY_INTERFACE_INFO Struct Reference

#include <TYDefs.h>

Collaboration diagram for TY_INTERFACE_INFO:



Public Attributes

- char name [32]
- char id [32]
- TY_INTERFACE_TYPE type
- char reserved [4]
- · TY DEVICE NET INFO netInfo

4.37.1 Detailed Description

See also

TYGetInterfaceList

Definition at line 831 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.38 TY_LASER_PARAM Struct Reference

Public Attributes

- uint32 t idx
- uint32_t en
- · uint32_t power

4.38.1 Detailed Description

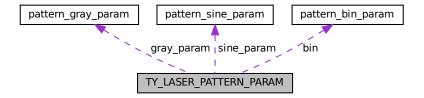
Definition at line 1219 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.39 TY_LASER_PATTERN_PARAM Struct Reference

 $Collaboration\ diagram\ for\ TY_LASER_PATTERN_PARAM:$



Public Attributes

```
    uint32_t img_index
    uint32_t type
    union {
        uint8_t payload [512+16]
        pattern_sine_param sine_param
        pattern_gray_param gray_param
        pattern_bin_param bin
}
```

4.39.1 Detailed Description

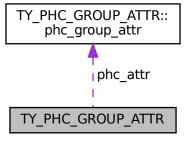
Definition at line 1091 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.40 TY_PHC_GROUP_ATTR Struct Reference

Collaboration diagram for TY_PHC_GROUP_ATTR:



Classes

struct phc_group_attr

Public Attributes

- uint32_t offset
- uint32_t size
- struct TY_PHC_GROUP_ATTR::phc_group_attr phc_attr [16]

4.40.1 Detailed Description

Definition at line 1051 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.41 TY_PIXEL_COLOR_DESC Struct Reference

Public Attributes

- int16 t x
- int16_t y
- uint8_t bgr_ch1
- uint8_t bgr_ch2
- uint8_t bgr_ch3
- uint8_t rsvd

4.41.1 Detailed Description

Definition at line 20 of file TYCoordinateMapper.h.

The documentation for this struct was generated from the following file:

· TYCoordinateMapper.h

4.42 TY_PIXEL_DESC Struct Reference

Public Attributes

- int16_t x
- int16_t **y**
- uint16_t depth
- uint16_t rsvd

4.42.1 Detailed Description

Definition at line 12 of file TYCoordinateMapper.h.

The documentation for this struct was generated from the following file:

• TYCoordinateMapper.h

30 Class Documentation

4.43 TY_TEMP_DATA Struct Reference

Public Attributes

- uint32 t id
- char **name** [16]
- char temp [16]
- char **desc** [16]

4.43.1 Detailed Description

Definition at line 1297 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.44 TY TOF FREQ Struct Reference

Public Attributes

- uint32_t freq1
- uint32_t freq2

4.44.1 Detailed Description

Definition at line 1206 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.45 TY_TRIGGER_PARAM Struct Reference

Public Attributes

- TY_TRIGGER_MODE mode
- int8_t fps
- int8_t rsvd

4.45.1 Detailed Description

Definition at line 989 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.46 TY_TRIGGER_PARAM_EX Struct Reference

Public Attributes

```
TY_TRIGGER_MODE mode
union {
    struct {
        int8_t fps
        int8_t duty
        int32_t laser_stream
        int32_t led_stream
        int32_t led_expo
        int32_t led_gain
    }
    struct {
        int32_t ir_gain [2]
    }
    int32_t rsvd [32]
};
```

4.46.1 Detailed Description

Definition at line 997 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.47 TY_TRIGGER_TIMER_LIST Struct Reference

Public Attributes

```
uint64_t start_time_usuint32_t offset_us_countuint32_t offset_us_list [50]
```

4.47.1 Detailed Description

Definition at line 1020 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

32 Class Documentation

4.48 TY_TRIGGER_TIMER_PERIOD Struct Reference

Public Attributes

- uint64_t start_time_us
- uint32_t trigger_count
- uint32_t period_us

4.48.1 Detailed Description

Definition at line 1028 of file TYDefs.h.

The documentation for this struct was generated from the following file:

· TYDefs.h

4.49 TY_VECT_3F Struct Reference

Public Attributes

- float x
- float y
- float z

4.49.1 Detailed Description

Definition at line 921 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.50 TY_VERSION_INFO Struct Reference

Public Attributes

- int32_t major
- int32_t minor
- int32_t patch
- int32_t reserved

4.50.1 Detailed Description

Definition at line 803 of file TYDefs.h.

The documentation for this struct was generated from the following file:

• TYDefs.h

4.51 TYEnumEntry Struct Reference

Public Attributes

- int32_t value
- char name [64]
- char tooltip [512]
- char description [512]
- char displayName [512]

4.51.1 Detailed Description

Definition at line 16 of file TYParameter.h.

The documentation for this struct was generated from the following file:

· TYParameter.h

4.52 z_stream_s Struct Reference

Public Attributes

- z_const Bytef * next_in
- uInt avail_in
- uLong total_in
- Bytef * next_out
- uInt avail_out
- uLong total out
- z_const char * msg
- struct internal_state FAR * state
- · alloc func zalloc
- free_func zfree
- · voidpf opaque
- int data_type
- · uLong adler
- · uLong reserved

4.52.1 Detailed Description

Definition at line 86 of file zlib.h.

The documentation for this struct was generated from the following file:

· zlib.h

34 Class Documentation

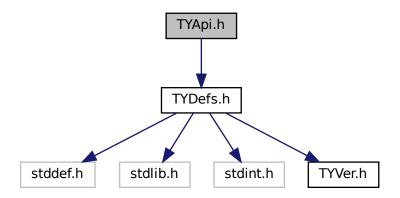
Chapter 5

File Documentation

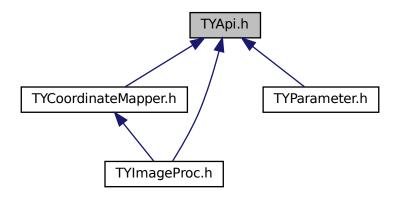
5.1 TYApi.h File Reference

TYApi.h includes camera control and data receiving interface, which supports configuration for image resolution, frame rate, exposure time, gain, working mode,etc.

#include "TYDefs.h"
Include dependency graph for TYApi.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef void(* TY_EVENT_CALLBACK) (TY_EVENT_INFO *, void *userdata)
- typedef void(* TY_IMU_CALLBACK) (TY_IMU_DATA *, void *userdata)

Functions

- TY_CAPI _TYInitLib (void)
- TY_CAPI TYLibVersion (TY_VERSION_INFO *version)

Get current library version.

• TY_EXTC const TY_EXPORT char *TY_STDC TYErrorString (TY_STATUS errorID)

Get error information.

• TY_CAPI TYDeinitLib (void)

Deinit this library.

• TY_CAPI TYSetLogLevel (TY_LOG_LEVEL lvl)

Set log level.

• TY_CAPI TYSetLogPrefix (const char *prefix)

set log prefix

• TY_CAPI TYAppendLogToFile (const char *filePath, TY_LOG_LEVEL lvl)

Append log to specified file.

• TY_CAPI TYRemoveLogFile (const char *filePath)

Remove log file.

- TY_CAPI TYAppendLogToServer (const char *protocol, const char *ip, uint16_t port, TY_LOG_LEVEL lvl)

 Append log to Tcp/Udp server.
- TY_CAPI TYRemoveLogServer (const char *protocol, const char *ip, uint16_t port)

Remove log server.

• TY_CAPI TYUpdateInterfaceList (void)

Update current interfaces. call before TYGetInterfaceList.

TY_CAPI TYGetInterfaceNumber (uint32_t *pNumIfaces)

Get number of current interfaces.

TY_CAPI TYGetInterfaceList (TY_INTERFACE_INFO *plfaceInfos, uint32_t bufferCount, uint32_t *filled ← Count)

Get interface info list.

• TY CAPI TYHasInterface (const char *ifaceID, bool *value)

Check if has interface.

TY_CAPI TYOpenInterface (const char *ifaceID, TY_INTERFACE_HANDLE *outHandle)

Open specified interface.

TY CAPI TYCloseInterface (TY INTERFACE HANDLE ifaceHandle)

Close interface.

• TY_CAPI TYUpdateDeviceList (TY_INTERFACE_HANDLE ifaceHandle)

Update current connected devices.

TY_CAPI TYUpdateAllDeviceList (void)

Update current connected devices.

 $\bullet \ \ TY_CAPI\ TYGetDeviceNumber\ (TY_INTERFACE_HANDLE\ if ace Handle,\ uint 32_t\ *deviceNumber)$

Get number of current connected devices.

TY_CAPI TYGetDeviceList (TY_INTERFACE_HANDLE ifaceHandle, TY_DEVICE_BASE_INFO *device ←
 Infos, uint32_t bufferCount, uint32_t *filledDeviceCount)

Get device info list.

• TY_CAPI TYHasDevice (TY_INTERFACE_HANDLE ifaceHandle, const char *deviceID, bool *value)

Check whether the interface has the specified device.

TY_CAPI TYOpenDevice (TY_INTERFACE_HANDLE ifaceHandle, const char *deviceID, TY_DEV_HANDLE *outDeviceHandle, TY_FW_ERRORCODE *outFwErrorcode=NULL)

Open device by device ID.

 TY_CAPI TYOpenDeviceWithIP (TY_INTERFACE_HANDLE ifaceHandle, const char *IP, TY_DEV_HANDLE *deviceHandle)

Open device by device IP, useful when a device is not listed.

• TY_CAPI TYGetDeviceInterface (TY_DEV_HANDLE hDevice, TY_INTERFACE_HANDLE *plface)

Get interface handle by device handle.

 TY_CAPI TYForceDeviceIP (TY_INTERFACE_HANDLE ifaceHandle, const char *MAC, const char *newIP, const char *newNetMask, const char *newGateway)

Force a ethernet device to use new IP address, useful when device use persistent IP and cannot be found.

• TY CAPI TYCloseDevice (TY DEV HANDLE hDevice, bool reboot=false)

Close device by device handle.

TY_CAPI TYGetDeviceInfo (TY_DEV_HANDLE hDevice, TY_DEVICE_BASE_INFO *info)
 Get base info of the open device.

• TY_CAPI TYGetComponentIDs (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID *componentIDs) Get all components IDs.

• TY_CAPI TYGetEnabledComponents (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID *componentIDs)

Get all enabled components IDs.

• TY_CAPI TYEnableComponents (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentIDs) Enable components.

TY_CAPI TYDisableComponents (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentIDs)
 Disable components.

TY_CAPI TYGetFrameBufferSize (TY_DEV_HANDLE hDevice, uint32_t *bufferSize)

Get total buffer size of one frame in current configuration.

• TY CAPI TYEnqueueBuffer (TY DEV HANDLE hDevice, void *buffer, uint32 t bufferSize)

Enqueue a user allocated buffer.

TY_CAPI TYClearBufferQueue (TY_DEV_HANDLE hDevice)

Clear the internal buffer queue, so that user can release all the buffer.

• TY CAPI TYStartCapture (TY DEV HANDLE hDevice)

Start capture.

TY_CAPI TYStopCapture (TY_DEV_HANDLE hDevice)
 Stop capture.

• TY_CAPI TYSendSoftTrigger (TY_DEV_HANDLE hDevice)

Send a software trigger to capture a frame when device works in trigger mode.

 TY_CAPI TYRegisterEventCallback (TY_DEV_HANDLE hDevice, TY_EVENT_CALLBACK callback, void *userdata)

Register device status callback. Register NULL to clean callback.

 TY_CAPI TYRegisterImuCallback (TY_DEV_HANDLE hDevice, TY_IMU_CALLBACK callback, void *userdata)

Register imu callback. Register NULL to clean callback.

• TY_CAPI TYFetchFrame (TY_DEV_HANDLE hDevice, TY_FRAME_DATA *frame, int32_t timeout) Fetch one frame.

• TY_CAPI TYHasFeature (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, bool *value)

Check whether a component has a specific feature.

TY_CAPI TYGetFeatureInfo (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, TY_FEATURE_INFO *featureInfo)

Get feature info.

• TY_CAPI TYGetIntRange (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, TY_INT_RANGE *intRange)

Get value range of integer feature.

TY_CAPI TYGetInt (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, int32_t *value)

Get value of integer feature.

• TY_CAPI TYSetInt (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, int32_t value)

Set value of integer feature.

• TY_CAPI TYGetFloatRange (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, TY_FLOAT_RANGE *floatRange)

Get value range of float feature.

• TY_CAPI TYGetFloat (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, float *value)

Get value of float feature.

• TY_CAPI TYSetFloat (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, float value)

Set value of float feature.

• TY_CAPI TYGetEnumEntryCount (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY FEATURE ID featureID, uint32 t *entryCount)

Get number of enum entries.

 TY_CAPI TYGetEnumEntryInfo (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, TY_ENUM_ENTRY *entries, uint32_t entryCount, uint32_t *filledEntryCount)
 Get list of enum entries.

• TY_CAPI TYGetEnum (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, uint32_t *value)

Get current value of enum feature.

• TY_CAPI TYSetEnum (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, uint32 t value)

Set value of enum feature.

• TY_CAPI TYGetBool (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, bool *value)

Get value of bool feature.

• TY_CAPI TYSetBool (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, bool value)

Set value of bool feature.

• TY_CAPI TYGetStringLength (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY FEATURE ID featureID, uint32 t *size)

Get internal buffer size of string feature.

• TY_CAPI TYGetString (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, char *buffer, uint32 t bufferSize)

Get value of string feature.

• TY_CAPI TYSetString (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, const char *buffer)

Set value of string feature.

• TY_CAPI TYGetStruct (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, void *pStruct, uint32_t structSize)

Get value of struct.

• TY_CAPI TYSetStruct (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, void *pStruct, uint32_t structSize)

Set value of struct.

• TY_CAPI TYGetByteArraySize (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY FEATURE ID featureID, uint32 t *pSize)

Get the size of specified byte array zone.

TY_CAPI TYGetByteArray (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, uint8 t *pBuffer, uint32 t bufferSize)

Read byte array from device.

• TY_CAPI TYSetByteArray (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, const uint8_t *pBuffer, uint32_t bufferSize)

Write byte array to device.

 TY_CAPI TYGetByteArrayAttr (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_ID featureID, TY_BYTEARRAY_ATTR *pAttr)

Write byte array to device.

 TY_CAPI TYGetDeviceFeatureNumber (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, uint32_t *size)

Get the size of device features.

• TY_CAPI TYGetDeviceFeatureInfo (TY_DEV_HANDLE hDevice, TY_COMPONENT_ID componentID, TY_FEATURE_INFO *featureInfo, uint32_t entryCount, uint32_t *filledEntryCount)

Get the all features by comp id.

• TY CAPI TYGetDeviceXMLSize (TY DEV HANDLE hDevice, uint32 t *size)

Get the Device xml size.

TY_CAPI TYGetDeviceXML (TY_DEV_HANDLE hDevice, char *xml, const uint32_t in_size, uint32_t *out
 — size)

Get the Device xml string.

5.1.1 Detailed Description

TYApi.h includes camera control and data receiving interface, which supports configuration for image resolution, frame rate, exposure

time, gain, working mode,etc.

5.1.2 Function Documentation

5.1.2.1 TYAppendLogToFile()

Append log to specified file.

Parameters

in	filePath	Path to the log file.
in <i>IVI</i>		Log level.

Return values

TY_STATUS_OK	Succeed.				
TY_STATUS_ERROR	Failed to add file				
	Suggestions: Please check if the file path is correct and if you have permiss	ion t	o writ	e t	.0

5.1.2.2 TYAppendLogToServer()

Append log to Tcp/Udp server.

Parameters

in	protocol	Protocol of the server, "tcp" or "udp".
in	ip	IP address of the server.
in	port	Port of the server.
in	lvl	Log level.

TY_STATUS_OK	Succeed.
TY_STATUS_ERROR	Failed to add server
	Suggestions: Please check if the ip and port are correct
TY_STATUS_INVALID_PARAMETER	Unsupported protocol
	Suggestions: Unsupported protocol, please use tcp or udp

5.1.2.3 TYClearBufferQueue()

Clear the internal buffer queue, so that user can release all the buffer.

Parameters

in hDevice Device handle.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYClearBufferQueue called with invalid device handle	
	Suggestions: Please check device handle	
	Like this: TYClearBufferOueue(hDevice);	ļ
	^ is invalid	ļ
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct han 2.Memory in stack to store handle data is corrupted	dle
	3.After getting handle, you updated device list by calli	ng TYUpdateD
TY_STATUS_BUSY	Device is capturing.	

5.1.2.4 TYCloseDevice()

```
TY_CAPI TYCloseDevice (

TY_DEV_HANDLE hDevice,

bool reboot = false )
```

Close device by device handle.

Parameters

in	hDevice	Device handle.
in	reboot	Reboot device after close.

TY_STATUS_OK	Succeed.
--------------	----------

Return values

TY_STATUS_INVALID_HANDLE	TYCloseDevice called with invalid device handle	
ļ	Suggestions:	
	Please check device handle	
	Like this:	
	TYCloseDevice(hDevice, reboot);	
	^ is invalid	
	The hDevice parameter you input is not recorded	
ļ	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handle
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated device list by ca	lling TYUpdateD
TY_STATUS_TIMEOUT	Failed to close device	
	Suggestions:	
	Possible reasons:	
	1.Network communication is abnormal, please check whe	ther the networ
	1. Notwork communication to approximately product shows and	chief elle heelet
TY_STATUS_DEVICE_ERROR	Failed to close device	
	Suggestions:	
	Possible reasons:	
	1.Camera device is abnormal and cannot be closed normal	allv
	1. Camera device is abnormal and cannot be crosed norm	arry.
TY STATUS IDLE	Device has been closed.	

5.1.2.5 TYCloseInterface()

```
TY_CAPI TYCloseInterface (
           TY_INTERFACE_HANDLE ifaceHandle )
```

Close interface.

Parameters

in	ifaceHandle	Interface to be closed.
	nacci ianaic	interiace to be closed.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	TYInitLib not called.	
TY_STATUS_INVALID_INTERFACE	TYCloseInterface called with invalid interface handle	
	Suggestions: Please check interface handle Like this: TYCloseInterface(ifaceHandle); ^ is invalid The ifaceHandle parameter you input is not recorded Possible reasons: 1.TYOpenInterface failed to open interface and get 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated interface list	d

5.1.2.6 TYDeinitLib()

```
TY_CAPI TYDeinitLib ( void )
```

Deinit this library.

Return values

```
TY_STATUS_OK Succeed.
```

5.1.2.7 TYDisableComponents()

Disable components.

Parameters

in	hDevice	Device handle.
in	componentIDs	Components to be disabled.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYDisableComponents called with invalid device handle
	Suggestions: Please check device handle Like this: TYDisableComponents(hDevice, componentIDs); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_PARAMETER	Invalid component IDs
	Suggestions: Please check componentIDs parameter Like this: TYDisableComponents(hDevice, componentIDs); ^ is invalid componentIDs should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr

Some components specified by componentIDs are invalid.

TY_STATUS_INVALID_COMPONENT

Return values

TY_STATUS_BUSY	Camera device is capturing	
	Suggestions: Please call TYEnableComponents when the camera dev Like this: TYStopCapture(hDevice); TYDisableComponents(hDevice, componentIDs);	ice is stopped

See also

TY_DEVICE_COMPONENT_LIST

5.1.2.8 TYEnableComponents()

```
TY_CAPI TYEnableComponents (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentIDs )
```

Enable components.

Parameters

in	hDevice	Device handle.
in	componentIDs	Components to be enabled.

TY_STATUS_INVALID_COMPONENT

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYEnableComponents called with invalid device handle	
	Suggestions: Please check device handle Like this: TYEnableComponents(hDevice, componentIDs); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get corn 2.Memory in stack to store handle data is corrupt 3.After getting handle, you updated device list h	ed
TY_STATUS_INVALID_PARAMETER	Invalid component IDs	
	Suggestions: Please check componentIDs parameter Like this: TYEnableComponents(hDevice, componentIDs); ^ is invalid componentIDs should be the value returned by TYGeto You can also view the components of the camera by	-

Some components specified by componentIDs are invalid.

Return values

TY_STATUS_BUSY	Camera device is capturing	
	Suggestions: Please call TYEnableComponents when the camera dev. Like this: TYStopCapture(hDevice); TYEnableComponents(hDevice, componentIDs);	ice is stopped

See also

TY_DEVICE_COMPONENT_LIST

5.1.2.9 TYEnqueueBuffer()

Enqueue a user allocated buffer.

Parameters

in	hDevice	Device handle.
in	buffer	Buffer to be enqueued.
in	bufferSize	Size of the input buffer.

Return values

Tiotain values		
TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYEnqueueBuffer called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYEnqueueBuffer(hDevice, buffer, bufferSize);	
	^ is invalid	
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handle
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated device list by ca	lling TYUpdateD
TY_STATUS_NULL_POINTER	TYEnqueueBuffer called with NULL pointer	
	Suggestions:	
	Please check your code	

TYEnqueueBuffer(hDevice, buffer, bufferSize);

^ is NULL

Like this:

Return values

TY_STATUS_WRONG_SIZE	TYEnqueueBuffer called with wrong size	
	Suggestions:	
	Please check your code	
	Like this:	
	TYEnqueueBuffer(hDevice, buffer, bufferSize);	
	^ is 0 or negative v	alue
TV 07.17.10 TV 15.01.7		
TY_STATUS_TIMEOUT	Failed to enqueue frame buffer	
	Suggestions:	
	Possible reasons:	
	1.Network communication is abnormal, please check whet	her the networ
TY_STATUS_DEVICE_ERROR	Failed to enqueue frame buffer	
	Suggestions:	
	Possible reasons:	
	1.Camera device is abnormal and cannot get the frame b	uffer size.

5.1.2.10 TYErrorString()

Get error information.

Parameters

in <i>errorID</i> Error	id.
-------------------------	-----

Return values

Error	string.

5.1.2.11 TYFetchFrame()

```
TY_CAPI TYFetchFrame (

TY_DEV_HANDLE hDevice,

TY_FRAME_DATA * frame,

int32_t timeout )
```

Fetch one frame.

Parameters

in	hDevice	Device handle.
out	frame	Frame data to be filled.
in	timeout	Timeout in milliseconds. <0 for infinite.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYFetchFrame called with invalid device handle	
	Suggestions: Please check device handle Like this: TYFetchFrame(hDevice, pFrame, timeout); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by ca	
TY_STATUS_NULL_POINTER	TYFetchFrame called with NULL pointer	
	Suggestions: Please check your code Like this: TYFetchFrame(hDevice, pFrame, timeout); ^ is NULL	
TY_STATUS_IDLE	Camera device is not started	
	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYFetchFrame(hDevice, pFrame, timeout);	
TY_STATUS_WRONG_MODE	Callback has been registered, this function is disabled.	
TY_STATUS_TIMEOUT	Failed to get frame	
	Suggestions: Possible reasons: 1.Camera device is abnormal and cannot get frame. 2.Network communication is abnormal, please check whe 3.Timeout, frame acquisition timeout	ther the netwo

5.1.2.12 TYForceDeviceIP()

```
TY_CAPI TYForceDeviceIP (

TY_INTERFACE_HANDLE ifaceHandle,

const char * MAC,

const char * newIP,

const char * newNetMask,

const char * newGateway )
```

Force a ethernet device to use new IP address, useful when device use persistent IP and cannot be found.

Parameters

in	ifaceHandle	Interface handle.
in	MAC	Device MAC, should be "xx:xx:xx:xx:xx:xx".
in	newIP	New IP.
in	newNetMask	New subnet mask.
in	newGateway	New gateway.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.
TY_STATUS_INVALID_INTERFACE	TYForceDeviceIP called with invalid interface handle
	Suggestions: Please check interface handle Like this: TYForceDeviceIP(ifaceHandle, MAC, newIP, newNetMask, newGateway) ^ is invalid The ifaceHandle parameter you input is not recorded Possible reasons: 1.TYOpenInterface failed to open interface and get correct handl 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated interface list by calling TY
TY_STATUS_WRONG_TYPE	TYForceDeviceIP called with invalid interface type
	Suggestions: Please check interface type Usually you can get interface information by calling TYGetInterface You can use TYIsNetworkInterface to check the interface type Only network interfaces can call TYForceDeviceIP Like this: TY_INTERFACE_INFO info; uint32_t num; TYGetInterfaceList(&info, 1, #); if(TYIsNetworkInterface(info[0].type)) { TY_INTERFACE_HANDLE hIface; TYOpenInterface(info[0].id, &hIface); TYForceDeviceIP(hIface, MAC, newIP, newNetMask, newGateway); }
TY_STATUS_NULL_POINTER	TYForceDeviceIP called with NULL pointer
	Suggestions: Please check your code Like this: TYForceDeviceIP(ifaceHandle, MAC, newIP, newNetMask, newGateway) ^ or ^ or ^ or ^ is NULL
TY_STATUS_INVALID_PARAMETER	Invalid MAC address:
	Suggestions: Please check MAC parameter Like this: TYForceDeviceIP(ifaceHandle, MAC, newIP, newNetMask, newGateway) ^ is invalid MAC address should be six bytes of hexadecimal separated by colons For example: 00:11:22:aa:bb:cc
TY_STATUS_TIMEOUT	Failed to force set IP
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the new communication is abnormal.
TY_STATUS_DEVICE_ERROR	Failed to force set IP
	Suggestions: Possible reasons: 1.New IP, NetMask, Gateway are incorrect, camera device refuses

^ type mismatch

The feature type you entered does not match. You can use TYFeature

5.1.2.13 TYGetBool()

```
TY_CAPI TYGetBool (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

bool * value )
```

Get value of bool feature.

Parameters

in	hDevice	Device handle.
in <i>componentID</i>		Component ID.
in	featureID	Feature ID.
out	value	Bool value.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetBool called with invalid device handle
	Suggestions:
	Please check device handle
	Like this: TYGetBool(hDevice, componentID, featureID, pValue);
	^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted
	3. After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this: TYGetBool(hDevice, componentID, featureID, pValue);
	^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the x
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions:
	Please check featureID parameter
	Like this: TYGetBool(hDevice, componentID, featureID, pValue);
	^ is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGet
	You can also view the features of the camera device by obtaining
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions:
	Please check the feature type
	Like this:
	TYGetBool(hDevice, componentID, featureID, pValue);

Return values

	TY_STATUS_NULL_POINTER	TYGetBool called with NULL pointer	1
		Suggestions: Please check your code Like this: TYGetBool(hDevice, componentID, featureID, pValue ^ is N	
	TY_STATUS_TIMEOUT	Failed to get bool feature	ı
		Suggestions: Possible reasons: 1.Network communication is abnormal, please check	whether the no
Ī	TY_STATUS_DEVICE_ERROR	Failed to get bool feature	1
		Suggestions: Possible reasons: 1.The feature of the camera device is not availab 2.Camera device is abnormal and cannot get bool f	_

5.1.2.14 TYGetByteArray()

Read byte array from device.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out <i>pBuffer</i>		Byte buffer.
in	bufferSize	Size of buffer.

TY_STATUS_INVALID_HANDLE TYGetByteArray called with invalid device handle	
Suggestions: Please check device handle Like this: TYGetByteArray(hDevice, componentID, featureID, buffer, k ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct hand 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling	ndle

Return values

TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetByteArray(hDevice, componentID, featureID, buffer, buffer ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetByteArray(hDevice, componentID, featureID, buffer, buffer ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGe
	You can also view the features of the camera device by obtaining
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetByteArray(hDevice, componentID, featureID, buffer, buffer ^ type mismatch The feature type you entered does not match. You can use TYFeatu
TY_STATUS_NULL_POINTER	TYGetByteArray called with NULL pointer
	Suggestions: Please check your code Like this: TYGetByteArray(hDevice, componentID, featureID, pBuffer, buffe is NULL
TY_STATUS_WRONG_SIZE	Array size mismatch
	Suggestions: Please check the array size Like this: TYGetByteArray(hDevice, componentID, featureID, buffer, buffer ^ is i The array size you entered does not match
TY_STATUS_TIMEOUT	Failed to get byte array feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the
TY_STATUS_DEVICE_ERROR	Failed to get byte array feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not important and cannot get byte array feature

5.1.2.15 TYGetByteArrayAttr()

```
TY_COMPONENT_ID componentID,
TY_FEATURE_ID featureID,
TY_BYTEARRAY_ATTR * pAttr )
```

Write byte array to device.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	pAttr	Byte array attribute to be filled.

TY_STATUS_INVALID_HANDLE	TYGetByteArrayAttr called with invalid device handle
	Gumanti ann
	Suggestions:
	Please check device handle
	Like this:
	TYGetByteArrayAttr(hDevice, componentID, featureID, pAttr); ^ is invalid
	The hDevice parameter you input is not recorded Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2.Memory in stack to store handle data is corrupted
	3.After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetByteArrayAttr(hDevice, componentID, feature D, pAttr); ^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the x
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions:
	Please check featureID parameter
	Like this:
	' is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGet
	rou can also view the reatures of the camera device by obtaining
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions:
	Please check the feature type
	Please check the feature type Like this:
	Please check the feature type
	You entered an invalid featureID parameter You can get a list of features of the camera dev You can also view the features of the camera dev The feature is not writable.

Return values

TY_STATUS_NULL_POINTER	TYGetByteArrayAttr called with NULL pointer	l
	Suggestions: Please check your code Like this: TYGetByteArrayAttr(hDevice, componentID, feature)	ID, pAttr); ^ is NULL

5.1.2.16 TYGetByteArraySize()

```
TY_CAPI TYGetByteArraySize (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

uint32_t * pSize )
```

Get the size of specified byte array zone.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	pSize	Size of specified byte array zone.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetByteArraySize called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetByteArraySize(hDevice, componentID, feature D, pSize); ^ is invalid
	The hDevice parameter you input is not recorded Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2.Memory in stack to store handle data is corrupted
	3.After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetByteArraySize(hDevice, componentID, feature D, pSize);
	^ is invalid

componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xi

Return values

TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetByteArraySize(hDevice, componentID, featureID, pSize); ^ is invalid You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYG You can also view the features of the camera device by obtaining
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetByteArraySize(hDevice, componentID, featureID, pSize); ^ type mismatch The feature type you entered does not match. You can use TYFeat
TY_STATUS_NULL_POINTER	TYGetByteArraySize called with NULL pointer
	Suggestions: Please check your code Like this: TYGetByteArraySize(hDevice, componentID, featureID, pSize); ^ is NULI

5.1.2.17 TYGetComponentIDs()

Get all components IDs.

Parameters

in	hDevice	Device handle.
out	componentIDs	All component IDs this device has. (bit flag).

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetComponentIDs called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetComponentIDs(hDevice, outComponentIDs);
	^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct
	2.Memory in stack to store handle data is corrupted
	3.After getting handle, you updated device list by ca
1	

Return values

TY_STATUS_NULL_POINTER	TYGetComponentIDs called with NULL pointer	
	Suggestions: Please check your code Like this: TYGetComponentIDs(hDevice, outComponentIDs); ^ is NULL	

See also

TY_DEVICE_COMPONENT_LIST

5.1.2.18 TYGetDeviceFeatureInfo()

```
TY_CAPI TYGetDeviceFeatureInfo (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_INFO * featureInfo,

uint32_t entryCount,

uint32_t * filledEntryCount)
```

Get the all features by comp id.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
out	featureInfo	Output feature info.
in	entryCount	Array size of input parameter "featureInfo".
out	filledEntryCount	Number of filled featureInfo.

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYGetDeviceFeatureInfo called with invalid device handle	
	Suggestions: Please check device handle Like this: TYGetDeviceFeatureInfo(hDevice, componentID, feature ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by o	t handle

Return values

TY STATUS INVALID COMPONENT	Invalid component ID	
	·	
	Suggestions:	
	Please check componentID parameter	
	Like this:	
	TYGetDeviceFeatureInfo(hDevice, componentID, feat ^ is invalid	lureinio, entry
	componentID should be the value returned by TYGetCo	mponentIDs
	You can also view the components of the camera by o	btaining the xr
TY_STATUS_NULL_POINTER	TYGetDeviceFeatureInfo called with NULL pointer	
	Suggestions:	
	Please check your code	
	Like this:	
	TYGetDeviceFeatureInfo(hDevice, componentID, feat	ureInfo, entry(
	^	or
TY_STATUS_TIMEOUT	Failed to get feature info	
	Suggestions:	
	Possible reasons:	
	1.Network communication is abnormal, please check	whether the ne
TY STATUS DEVICE ERROR	Failed to get feature info	
	- Amount of government and	
	Suggestions:	
	Possible reasons:	
	 The feature of the camera device is not available Camera device is abnormal and cannot get feature 	-

5.1.2.19 TYGetDeviceFeatureNumber()

Get the size of device features.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
out	size	Size of all feature cnt.

TY_STATUS_OK	Succeed.

Return values

TY_STATUS_INVALID_HANDLE	TYGetDeviceFeatureNumber called with invalid device handle	
	Suggestions: Please check device handle Like this: TYGetDeviceFeatureNumber(hDevice, componentID, size); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling	
TY_STATUS_INVALID_COMPONENT	Invalid component ID	
	Suggestions: Please check componentID parameter Like this: TYGetDeviceFeatureNumber(hDevice, componentID, size); ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining t	
TY_STATUS_NULL_POINTER	TYGetDeviceFeatureNumber called with NULL pointer	
	Suggestions: Please check your code Like this: TYGetDeviceFeatureNumber(hDevice, componentID, size); ' is NULL	
TY_STATUS_TIMEOUT	Failed to get feature number	
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether t	the ne
TY_STATUS_DEVICE_ERROR	Failed to get feature number	
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not 2.Camera device is abnormal and cannot get feature number	impl∈

5.1.2.20 TYGetDeviceInfo()

Get base info of the open device.

Parameters

in	hDevice	Device handle.
out	info	Base info out.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetDeviceInfo called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetDeviceInfo(hDevice, info);
TY_STATUS_NULL_POINTER	TYGetDeviceInfo called with NULL pointer
	Suggestions: Please check your code Like this: TYGetDeviceInfo(hDevice, info); ^ is NULL

5.1.2.21 TYGetDeviceInterface()

Get interface handle by device handle.

Parameters

in	hDevice	Device handle.	
out	plface	Interface handle.	

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYGetDeviceInterface called with invalid device handle	
	<pre>Suggestions: Please check device handle Like this: TYGetDeviceInterface(hDevice, pIface);</pre>	
	^ is invalid The hDevice parameter you input is not recorded	
	Possible reasons: 1.TYOpenDevice failed to open device and get correct 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by ca	

Return values

TY_STATUS_NULL_POINTER	TYGetDeviceInterface called with NULL pointer
	Suggestions: Please check your code Like this: TYGetDeviceInterface(hDevice, pIface); ^ is NULL

5.1.2.22 TYGetDeviceList()

```
TY_CAPI TYGetDeviceList (
            TY_INTERFACE_HANDLE ifaceHandle,
            TY_DEVICE_BASE_INFO * deviceInfos,
            uint32_t bufferCount,
            uint32_t * filledDeviceCount )
```

Get device info list.

Parameters

in	ifaceHandle	Interface handle.
out	deviceInfos	Device info array to be filled.
in	bufferCount	Array size of deviceInfos.
out	filledDeviceCount	Number of filled TY_DEVICE_BASE_INFO.

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	TYInitLib not called.	
TY_STATUS_INVALID_INTERFACE	TYGetDeviceList called with invalid interface handle	
	Suggestions:	
	Please check interface handle	
	Like this:	
	TYGetDeviceList(ifaceHandle, pDeviceInfos, bufferCo ^ is invalid	unt, pFilledDe
	The ifaceHandle parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenInterface failed to open interface and get	
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated interface list	by calling TYU
TY STATUS NULL POINTER	TYGetDeviceList called with NULL pointer	
	1 1 0 0 1 2 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 1	
	Suggestions:	
	Please check your code	
	Like this:	
	TYGetDeviceList(ifaceHandle, pDeviceInfos, bufferCo	unt, pFilledCo
	^ is NULL or ^ is 0	or ^ is NULL

5.1.2.23 TYGetDeviceNumber()

Get number of current connected devices.

Parameters

in	ifaceHandle	Interface handle.
out	deviceNumber	Number of connected devices.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	TYInitLib not called.	
TY_STATUS_INVALID_INTERFACE	TYGetDeviceNumber called with invalid interface handle	
	Suggestions: Please check interface handle Like this: TYGetDeviceNumber(ifaceHandle, pDeviceNumber); ^ is invalid The ifaceHandle parameter you input is not recorded Possible reasons: 1.TYOpenInterface failed to open interface and get 2.Memory in stack to store handle data is corrupte 3.After getting handle, you updated interface list	d
TY_STATUS_NULL_POINTER	TYGetDeviceNumber called with NULL pointer Suggestions: Please check your code Like this: TYGetDeviceNumber(ifaceHandle, deviceNumber); ^ is NULL	

5.1.2.24 TYGetDeviceXML()

Get the Device xml string.

Parameters

in	hDevice	Device handle.
in	xml	The buffer to store xml
in	in_size	The size buffer
out	out_size	The actual size write in buffer

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	Not call TYInitLib	
TY_STATUS_WRONG_SIZE	XML buffer size is not enough	
	Suggestions:	
	Suggestions: XML buffer size is not enough	
	Like this:	
	TYGetDeviceXML(hDevice, xml, in_size, out_size);	
	^ is invalid	
	XML buffer size is not enough, please use TYGetDeviceXM	LSize to get th
TY_STATUS_INVALID_HANDLE	TYGetDeviceXML called with invalid device handle	
77_077.1.00	Tradiborios militaria donos narias	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYGetDeviceXML(hDevice, xml, in_size, out_size);	
	^ is invalid	
	The hDevice parameter you input is not recorded Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handle
	2. Memory in stack to store handle data is corrupted	nandie
	3. After getting handle, you updated device list by ca	lling TYUpdateD
		, <u>.</u>
TY_STATUS_NULL_POINTER	TYGetDeviceXML called with NULL pointer	
	Suggestions:	
	Please check your code	
	Like this:	
	TYGetDeviceXML(hDevice, xml, in_size, out_size);	
	^ or ^ is NULL	

5.1.2.25 TYGetDeviceXMLSize()

Get the Device xml size.

Parameters

in	hDevice	Device handle.
out	size	The size of device xml string

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	Not call TYInitLib

Return values

TY_STATUS_INVALID_HANDLE	TYGetDeviceXMLSize called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYGetDeviceXMLSize(hDevice, size);	
	^ is invalid	
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct handle	
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated device list by ca	lling TYUpdateD
TY STATUS NULL POINTER	TYGetDeviceXMLSize called with NULL pointer	
	- 1 - 00 12 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	
	Suggestions:	
	Please check your code	
	Like this:	
	TYGetDeviceXMLSize(hDevice, size);	
	^ is NULL	

5.1.2.26 TYGetEnabledComponents()

Get all enabled components IDs.

Parameters

in	hDevice	Device handle.
out	componentIDs	Enabled component IDs.(bit flag)

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYGetEnabledComponents called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	<pre>TYGetEnabledComponents(hDevice, componentIDs);</pre>	
^ is invalid		
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handle
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated device list by ca	lling TYUpdateD
TV OTATUO AUUL DOMITED	UD : AUU I	
TY_STATUS_NULL_POINTER	componentIDs is NULL.	

See also

TY_DEVICE_COMPONENT_LIST

5.1.2.27 TYGetEnum()

Get current value of enum feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	value	Enum value.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetEnum called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetEnum(hDevice, componentID, featureID, pValue);
	^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2. Memory in stack to store handle data is corrupted
	3. After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID: d
0,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetEnum(hDevice, componentID, featureID, pValue);
	^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the x
TV STATUS INVALID FEATURE	Invalid feature ID: d
TY_STATUS_INVALID_FEATURE	invalid leature ib. d
	Suggestions:
	Please check featureID parameter
	Like this:
	TYGetEnum(hDevice, componentID, featureID, pValue);
	^ is invalid
	We are the section of

You entered an invalid featureID parameter

You can get a list of features of the camera device through TYGet?
You can also view the features of the camera device by obtaining

Return values

TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type
	Like this:
	TYGetEnum(hDevice, componentID, featureID, pValue);
	The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYGetEnum called with NULL pointer
	Suggestions: Please check your code Like this:
	TYGetEnum(hDevice, componentID, featureID, pValue); ^ is NULL
TY_STATUS_TIMEOUT	Failed to get enum feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the ne
TY_STATUS_DEVICE_ERROR	Failed to get enum feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not imple 2.Camera device is abnormal and cannot get enum feature

5.1.2.28 TYGetEnumEntryCount()

```
TY_CAPI TYGetEnumEntryCount (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

uint32_t * entryCount )
```

Get number of enum entries.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	entryCount	Entry count.

TY_STATUS_OK	Succeed.

TY_STATUS_INVALID_HANDLE	TYGetEnumEntryCount called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetEnumEntryCount(hDevice, componentID, featureID, pEntryCount
	^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetEnumEntryCount(hDevice, componentID, featureID, pEntryCount
	^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetEnumEntryCount(hDevice, componentID, featureID, pEntryCount ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the component of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the c
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetEnumEntryCount(hDevice, componentID, featureID, pEntryCount ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYGetEnumEntryCount called with NULL pointer
	Suggestions: Please check your code Like this: TYCOTERUMENT TO COUNT (browning commonant ID footung ID pent ty Count
	TYGetEnumEntryCount(hDevice, componentID, featureID, pEntryCount

5.1.2.29 TYGetEnumEntryInfo()

```
TY_CAPI TYGetEnumEntryInfo (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

TY_ENUM_ENTRY * entries,

uint32_t entryCount,

uint32_t * filledEntryCount )
```

Get list of enum entries.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	entries	Output entries.
in	entryCount	Array size of input parameter "entries".
out	filledEntryCount	Number of filled entries.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetEnumEntryInfo called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetEnumEntryInfo(hDevice, componentID, featureID, entries, ent ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_COMPONENT	Invalid component ID: d
	Suggestions: Please check componentID parameter Like this: TYGetEnumEntryInfo(hDevice, componentID, featureID, entries, ent ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID: d
	Suggestions: Please check featureID parameter Like this: TYGetEnumEntryInfo(hDevice, componentID, featureID, entries, ent ' is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the state of the
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetEnumEntryInfo(hDevice, componentID, featureID, entries, entition of type mismatch) The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYGetEnumEntryInfo called with NULL pointer
	Suggestions: Please check your code Like this: TYGetEnumEntryInfo(hDevice, componentID, featureID, pEnumDescri

5.1.2.30 TYGetFeatureInfo()

```
TY_CAPI TYGetFeatureInfo (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

TY_FEATURE_INFO * featureInfo )
```

Get feature info.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	featureInfo	Feature info.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetFeatureInfo called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetFeatureInfo(hDevice, componentID, featureID, pFeatureInfo ^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted
	3. After getting handle, you updated device list by calling TYN
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetFeatureInfo(hDevice, componentID, featureID, pFeatureInfo
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the
TY_STATUS_INVALID_COMPONENT	Invalid feature ID
	Suggestions:
	Please check featureID parameter
	Like this:
	TYGetFeatureInfo(hDevice, componentID, featureID, pFeatureInfo ^ is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGe
	You can also view the features of the camera device by obtaining
TY_STATUS_NULL_POINTER	TYGetFeatureInfo called with NULL pointer
	Suggestions:
	Please check your code
	Like this:
	myggipal at the College to the control of the contr

TYGetFeatureInfo(hDevice, componentID, featureID, pFeatureInfo)

^ is NULL

5.1.2.31 TYGetFloat()

```
TY_CAPI TYGetFloat (
            TY_DEV_HANDLE hDevice,
            TY_COMPONENT_ID componentID,
            TY_FEATURE_ID featureID,
            float * value )
```

Get value of float feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	value	Float value.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetFloat called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetFloat(hDevice, componentID, featureID, pValue); ^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUp
	3.After getting nanure, you updated device fist by carring frop
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetFloat(hDevice, componentID, featureID, pValue); ^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the x
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Currentions
	Suggestions: Please check featureID parameter
	Like this:
	TYGetFloat(hDevice, componentID, featureID, pValue); ^ is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGet
	You can also view the features of the camera device by obtaining
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions:
	Please check the feature type
	Like this:
	TYCHELLAL (ADDITION TO THE TO FORTUNE TO THE TYPE TYPE TO THE TYPE TYPE TYPE TYPE TYPE TYPE TYPE TYP

^ type mismatch

TYGetFloat(hDevice, componentID, featureID, pValue);

The feature type you entered does not match. You can use TYFeature

TY_STATUS_NULL_POINTER	TYGetFloat called with NULL pointer Suggestions: Please check your code Like this: TYGetFloat(hDevice, componentID, featureID, pValue)	1 1	
TY_STATUS_TIMEOUT	Failed to get float feature Suggestions: Possible reasons: 1.Network communication is abnormal, please check	NULL k whether	the no
TY_STATUS_DEVICE_ERROR	Failed to get float feature Suggestions: Possible reasons: 1.The feature of the camera device is not available 2.Camera device is abnormal and cannot get float		: impl

5.1.2.32 TYGetFloatRange()

```
TY_CAPI TYGetFloatRange (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

TY_FLOAT_RANGE * floatRange )
```

Get value range of float feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	floatRange	Float range to be filled.

TY_STATUS_OK	Succeed.]
TY_STATUS_INVALID_HANDLE	TYGetFloatRange called with invalid device handle	
	Suggestions:	
	Please check device handle	ļ
	Like this:	ļ
	TYGetFloatRange(hDevice, componentID, featureID, ^ is invalid	pFloatRange);
	The hDevice parameter you input is not recorded	ļ
	Possible reasons:	
	1.TYOpenDevice failed to open device and get cor	rect handle
	2.Memory in stack to store handle data is corrup	ted
	3.After getting handle, you updated device list	y calling TYUpo

Return values

TY_STATUS_INVALID_COMPONENT	Invalid component ID	
	Suggestions: Please check componentID parameter Like this: TYGetFloatRange(hDevice, componentID, featureID, ^ is invalid componentID should be the value returned by TYGetCo	
	You can also view the components of the camera by a	±
TY_STATUS_INVALID_FEATURE	Invalid feature ID	
	Suggestions: Please check featureID parameter Like this: TYGetFloatRange(hDevice, componentID, featureID, ^ is inval: You entered an invalid featureID parameter You can get a list of features of the camera device You can also view the features of the camera device	d through TYGetI
TY_STATUS_WRONG_TYPE	Feature type mismatch	
	Suggestions: Please check the feature type Like this: TYGetFloatRange(hDevice, componentID, featureID, ^ type miss The feature type you entered does not match. You ca	natch
TY_STATUS_NULL_POINTER	TYGetFloatRange called with NULL pointer	
	Suggestions: Please check your code Like this: TYGetFloatRange(hDevice, componentID, featureID,	pFloatRange); ^ is NULL

5.1.2.33 TYGetFrameBufferSize()

Get total buffer size of one frame in current configuration.

Parameters

in	hDevice	Device handle.
out	bufferSize	Buffer size per frame.

TY_STATUS_OK	Succeed.

TY_STATUS_INVALID_HANDLE	TYGetFrameBufferSize called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetFrameBufferSize(hDevice, outSize); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpdateD
TY_STATUS_NULL_POINTER	TYGetFrameBufferSize called with NULL pointer
	Suggestions: Please check your code Like this: TYGetFrameBufferSize(hDevice, outSize); ^ is NULL
TY_STATUS_TIMEOUT	Failed to get frame buffer size
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the network
TY_STATUS_DEVICE_ERROR	Failed to get frame buffer size
	Suggestions: Possible reasons: 1.Camera device is abnormal and cannot get the frame buffer size.

5.1.2.34 TYGetInt()

Get value of integer feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	value	Integer value.

TY_STATUS_OK	Succeed.

TV STATUS INVALID HANDLE	TYGetInt called with invalid device handle
TY_STATUS_INVALID_HANDLE	
	Suggestions: Please check device handle
	Like this:
	TYGetInt(hDevice, componentID, featureID, pValue); ^ is invalid
	The hDevice parameter you input is not recorded Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetInt(hDevice, componentID, featureID, pValue);
	^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetInt(hDevice, componentID, featureID, pValue); ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the component of the camera device by obtaining the camera device by obtainin
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetInt(hDevice, componentID, featureID, pValue); ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYGetInt called with NULL pointer
	Suggestions: Please check your code Like this: TYGetInt(hDevice, componentID, featureID, pValue); ^ is NULL
TY_STATUS_TIMEOUT	Failed to get int feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the ne
TY_STATUS_DEVICE_ERROR	Failed to get int feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not imple 2.Camera device is abnormal and cannot get int feature

5.1.2.35 TYGetInterfaceList()

Get interface info list.

Parameters

out	plfaceInfos	Array of interface infos to be filled.
in	bufferCount	Array size of interface infos.
out	filledCount	Number of filled TY_INTERFACE_INFO.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	TYInitLib not called.	
TY_STATUS_NULL_POINTER	TYGetInterfaceList called with NULL pointer	
	Suggestions: Please check your code Like this: TYGetInterfaceList(pIfaceInfos, bufferCount, filledCount ^ or ^ is NULL	:);

5.1.2.36 TYGetInterfaceNumber()

```
TY_CAPI TYGetInterfaceNumber ( \mbox{uint32\_t} \ * \ p\mbox{NumIfaces} \ )
```

Get number of current interfaces.

Parameters

out	pNumlfaces	Number of interfaces.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.
TY_STATUS_NULL_POINTER	TYGetInterfaceNumber called with NULL pointer
	Suggestions: Please check your code Like this: TYGetInterfaceNumber(pNumIfaces); ^ is NULL

5.1.2.37 TYGetIntRange()

```
TY_CAPI TYGetIntRange (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

TY_INT_RANGE * intRange )
```

Get value range of integer feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	intRange	Integer range to be filled.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetIntRange called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYGetIntRange(hDevice, componentID, featureID, pintRange); ^ is invalid
	The hDevice parameter you input is not recorded Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2.Memory in stack to store handle data is corrupted
	3.After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYGetIntRange(hDevice, componentID, featureID, pintRange); ^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the x
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions:
	Please check featureID parameter
	Like this:
	TYGetIntRange(hDevice, componentID, featureID, pintRange); ^ is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGet

You can also view the features of the camera device by obtaining

TY_STATUS_WRONG_TYPE	Feature type mismatch	
	Suggestions:	
	Please check the feature type	
	Like this:	
	TYGetIntRange(hDevice, componentID, featureID, pl	IntRange);
	^ type mismat	ch
	The feature type you entered does not match. You ca	an use TYFeatu
TY_STATUS_NULL_POINTER	TYGetIntRange called with NULL pointer	
	Suggestions:	
	Please check your code	
	Like this:	
	TYGetIntRange(hDevice, componentID, featureID, p	IntRange); is NULL

5.1.2.38 TYGetString()

Get value of string feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	buffer	String buffer.
in	bufferSize	Size of buffer.

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYGetString called with invalid device handle	
	Suggestions: Please check device handle	
	Like this:	
	TYGetString(hDevice, componentID, featureID, buff ^ is invalid	er, bufferSize
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get core 2.Memory in stack to store handle data is corrupt	
	3. After getting handle, you updated device list b	

Return values

TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize) ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize) ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize) ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYGetString called with NULL pointer
	Suggestions: Please check your code Like this: TYGetString(hDevice, componentID, featureID, pBuffer, bufferSize ^ is NULL
TY_STATUS_TIMEOUT	Failed to get string feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the ne
TY_STATUS_DEVICE_ERROR	Failed to get string feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not imple 2.Camera device is abnormal and cannot get string feature

See also

TYGetStringLength

5.1.2.39 TYGetStringLength()

```
TY_FEATURE_ID featureID,
uint32_t * size )
```

Get internal buffer size of string feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	size	String length including '\0'.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetString called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize ^ is invalid The hDevice parameter you input is not recorded
	Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUp
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the x
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGet You can also view the features of the camera device by obtaining
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYGetString(hDevice, componentID, featureID, buffer, bufferSize ^ type mismatch The feature type you entered does not match. You can use TYFeatur
TY_STATUS_NULL_POINTER	TYGetStringLength called with NULL pointer
	Suggestions: Please check your code Like this: TYGetStringLength(hDevice, componentID, featureID, pLength); ^ is NULL

See also

TYGetString

5.1.2.40 TYGetStruct()

```
TY_CAPI TYGetStruct (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

void * pStruct,

uint32_t structSize )
```

Get value of struct.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	pStruct	Pointer of struct.
in	structSize	Size of input buffer pStruct.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYGetStruct called with invalid device handle
	Suggestions: Please check device handle Like this: TYGetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYGetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xm
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYGetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining t
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this:

TYGetStruct(hDevice, componentID, featureID, pStruct, structSize

The feature type you entered does not match. You can use TYFeature

^ type mismatch

Return values

TY_STATUS_NULL_POINTER	TYGetStruct called with NULL pointer	
	Suggestions: Please check your code Like this: TYGetStruct(hDevice, componentID, featureID, pSt ^ i	ruct, structSize s NULL
TY_STATUS_WRONG_SIZE	Struct size mismatch	-
	Suggestions: Please check the struct size Like this: TYGetStruct(hDevice, componentID, featureID, pSt The struct size you entered does not match	ruct, structSize ^ is inval
TY_STATUS_TIMEOUT	Failed to get struct feature	
	Suggestions: Possible reasons: 1.Network communication is abnormal, please chec	k whether the ne
TY_STATUS_DEVICE_ERROR	Failed to get struct feature]
	Suggestions: Possible reasons: 1.The feature of the camera device is not availa 2.Camera device is abnormal and cannot get struc	-

5.1.2.41 TYHasDevice()

Check whether the interface has the specified device.

Parameters

	in	ifaceHandle	Interface handle.
Ī	in	deviceID	Device ID string, can be get from TY_DEVICE_BASE_INFO.
Ī	out	value	True if the device exists.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.

TY_STATUS_INVALID_INTERFACE	TYHasDevice called with invalid interface handle	
	Suggestions: Please check interface handle Like this: TYHasDevice(ifaceHandle, deviceID, value); ^ is invalid The ifaceHandle parameter you input is not recorded Possible reasons: 1.TYOpenInterface failed to open interface and get 2.Memory in stack to store handle data is corrupte 3.After getting handle, you updated interface list	d
TY_STATUS_NULL_POINTER	TYHasDevice called with NULL pointer	
	Suggestions: Please check your code Like this: TYHasDevice(ifaceHandle, deviceID, value); ^ or ^ is NULL	

5.1.2.42 TYHasFeature()

```
TY_CAPI TYHasFeature (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

bool * value )
```

Check whether a component has a specific feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	value	Whether has feature.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYHasFeature called with invalid device handle
	Suggestions: Please check device handle Like this: TYHasFeature(hDevice, componentID, featureID, value);
	^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYU

Return values

TY_STATUS_INVALID_COMPONENT	Invalid component ID	
	Suggestions:	
	Please check componentID parameter	
	Like this:	
	TYHasFeature(hDevice, componentID, featureID, vai	ue);
	componentID should be the value returned by TYGetCo	mponentIDs
	You can also view the components of the camera by o	btaining the
TY_STATUS_NULL_POINTER	TYHasFeature called with NULL pointer	
	Suggestions:	
	Please check your code	
	Like this:	
	TYHasFeature(hDevice, componentID, featureID, va	ue);
	^ =	s NULL

5.1.2.43 TYHasInterface()

Check if has interface.

Parameters

in	ifaceID	Interface ID string, can be get from TY_INTERFACE_INFO.
out	value	True if the interface exists.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.
TY_STATUS_NULL_POINTER	TYHasInterface called with NULL pointer
	Suggestions: Please check your code Like this: TYHasInterface(ifaceID, value); ^ is NULL

See also

TYGetInterfaceList

5.1.2.44 TYLibVersion()

Get current library version.

Parameters

out version Version infomation to be filled.
--

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_NULL_POINTER	TYLibVersion called with NULL pointer
	Suggestions: Please check your code Like this: TYLibVersion(ver); ^ is NULL

5.1.2.45 TYOpenDevice()

Open device by device ID.

Parameters

in	ifaceHandle	Interface handle.
in	deviceID	Device ID string, can be get from TY_DEVICE_BASE_INFO.
out	outDeviceHandle	Handle of opened device. Valid only if TY_STATUS_OK or TY_FW_ERRORCODE returned.
out	outFwErrorcode	Firmware errorcode. Valid only if TY_FW_ERRORCODE returned.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.

TY_STATUS_INVALID_INTERFACE	TYOpenDevice called with invalid interface handle
	Suggestions: Please check interface handle Like this:
	TYOpenDevice(ifaceHandle, deviceID, outDeviceHandle, outFwErrord
	The ifaceHandle parameter you input is not recorded Possible reasons:
	<pre>1.TYOpenInterface failed to open interface and get correct handl 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated interface list by calling TY</pre>
TY_STATUS_NULL_POINTER	TYOpenDevice called with NULL pointer
	Suggestions: Please check your code Like this:
	TYOpenDevice(ifaceHandle, deviceID, outDeviceHandle, outFwErrord
TY_STATUS_INVALID_PARAMETER	TYOpenDevice called with invalid device ID: s
	Suggestions: Please check deviceID parameter Like this:
	TYOpenDevice(ifaceHandle, deviceID, outDeviceHandle, outFwErrorc
	Usually you get device information by calling TYUpdateDeviceList, and then open device by calling TYOpenDevice When your device online status changes); you may need to update device list again
TY_STATUS_BUSY	Failed to open device
	Suggestions: Possible reasons: 1.Camera is occupied, please check if other processes on this ma or other host machines are occupying the camera. If the camera 2.A third-party program is written into the camera, please conta
TY_STATUS_FIRMWARE_ERROR	Device opened successfully, but firmware error code is not 0
	Suggestions: Some functions of the device may have exceptions, please check the TY_FW_ERRORCODE outFwErrorcode; TYOpenDevice(ifaceHandle, deviceID, outDeviceHandle, &outFwErrorcoif(outFwErrorcode != 0) { parse_firmware_errcode(outFwErrorcode); }
TY_STATUS_DEVICE_ERROR	Failed to open device
	Suggestions: Possible reasons: 1.A third-party program is written into the camera, please conta 2.The camera IP address is not in the same network segment as th If the camera IP address is not in the same network segment as If there is a routing connection between your host and the cam Otherwise, you can modify the camera IP address or the host IP If you need to modify the camera IP address, please refer to S 3.Network communication is abnormal, please check whether the ne

5.1.2.46 TYOpenDeviceWithIP()

Open device by device IP, useful when a device is not listed.

Parameters

in	ifaceHandle	Interface handle.
in	IP	Device IP.
out	deviceHandle	Handle of opened device.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.
TY STATUS INVALID INTERFACE	TYOpenDeviceWithIP called with invalid interface handle
TI_STATOS_INVALID_INTERITACE	1 Topenbevicewithin called with invalid interlace handle
	Suggestions:
	Please check interface handle Like this:
	TYOpenDeviceWithIP(ifaceHandle, IP, outDeviceHandle); ^ is invalid
	The ifaceHandle parameter you input is not recorded Possible reasons:
	1.TYOpenInterface failed to open interface and get correct hand
	2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated interface list by calling 3
TY_STATUS_NULL_POINTER	TYOpenDeviceWithIP called with NULL pointer
	Suggestions:
	Please check your code
	Like this:
	TYOpenDeviceWithIP(ifaceHandle, IP, outDeviceHandle); ^ or^ is NULL
TY_STATUS_INVALID_PARAMETER	TYOpenDeviceWithIP called with invalid IP address
	Suggestions:
	Please check your code
	Like this:
	TYOpenDeviceWithIP(ifaceHandle, IP, outDeviceHandle); ^ is invalid
	A valid IP address should be like: 192.168.31.1
	Usually you get device information by calling TYUpdateDeviceList,
	and then open device by calling TYOpenDevice
	When your device online status changes, you may need to update device list again
TY_STATUS_BUSY	Failed to open device
	Suggestions:
	Possible reasons:
	1. Camera is occupied, please check if other processes on this many than the state of the state
	or other host machines are occupying the camera. If the camera. 2.A third-party program is written into the camera, please conf

Return values

TY_STATUS_DEVICE_ERROR	Failed to open device	
	Suggestions: Possible reasons:	
	1.A third-party program is written into the camer 2.The camera IP address cannot communicate with t	_
	3.Network communication is abnormal, please check	

5.1.2.47 TYOpenInterface()

Open specified interface.

Parameters

	in	ifaceID	Interface ID string, can be get from TY_INTERFACE_INFO.
Ī	out	outHandle	Handle of opened interface.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_NOT_INITED	TYInitLib not called.	
TY_STATUS_NULL_POINTER	TYOpenInterface called with NULL pointer	
	Suggestions: Please check your code Like this: TYOpenInterface(ifaceID, outHandle); ^ or ^ is NULL	
TY_STATUS_INVALID_INTERFACE	TYOpenInterface called with invalid interface ID	
	Suggestions: Please check ifaceID parameter Like this: TYOpenInterface(ifaceID, outHandle); ^ is invalid Usually you get interface information by calling TYU and then open interface by calling TYOpenInterface When your host interface (network or USB) changes); you may need to update interface list again	pdateInterfaceL

See also

TYGetInterfaceList

5.1.2.48 TYRegisterEventCallback()

Register device status callback. Register NULL to clean callback.

Parameters

in	hDevice	Device handle.
in	callback	Callback function.
in	userdata	User private data.

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYRegisterEventCallback called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYRegisterEventCallback(hDevice, callback, userdata); ^ is invalid	
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handlo
	2. Memory in stack to store handle data is corrupted	nandie
	3.After getting handle, you updated device list by ca	lling TYUpdat
TY_STATUS_BUSY	Device is capturing.	

5.1.2.49 TYRegisterImuCallback()

```
TY_CAPI TYRegisterImuCallback (

TY_DEV_HANDLE hDevice,

TY_IMU_CALLBACK callback,

void * userdata )
```

Register imu callback. Register NULL to clean callback.

Parameters

in	hDevice	Device handle.
in	callback	Callback function.
in	userdata	User private data.

TY_STATUS_OK Succeed.

Return values

TY_STATUS_INVALID_HANDLE	TYRegisterImuCallback called with invalid device handle	
	Suggestions: Please check device handle Like this: TYRegisterImuCallback(hDevice, callback, userdata); ^ is invalid	
	The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by ca	
TY_STATUS_BUSY	Device is capturing.	

5.1.2.50 TYRemoveLogFile()

Remove log file.

Parameters

in	filePath	Path to the log file.
----	----------	-----------------------

Return values

TY_STATUS_OK	Succeed.	
TY_STATUS_ERROR	Failed to remove file	
	Suggestions: Please check if the file path is correct	

5.1.2.51 TYRemoveLogServer()

Remove log server.

Parameters

in	protocol	Protocol of the server, "tcp" or "udp".
in	ip	IP address of the server.
in	port	Port of the server.

TY_STATUS_OK	Succeed.
TY_STATUS_ERROR	Failed to remove server
	Suggestions: Please check if the ip and port are correct
TY_STATUS_INVALID_PARAMETER	Unsupported protocol
	Suggestions: Unsupported protocol, please use tcp or udp

5.1.2.52 TYSendSoftTrigger()

Send a software trigger to capture a frame when device works in trigger mode.

Parameters

j	Ĺn	hDevice	Device handle.
---	----	---------	----------------

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYSendSoftTrigger called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYSendSoftTrigger(hDevice);	
	^ is invalid	
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	handle
	2. Memory in stack to store handle data is corrupted	. 1 1 ' mszrz 1
	3.After getting handle, you updated device list by d	alling TYUpdate
TV CTATUC INVALID FEATURE	Not assessed and twisters	
TY_STATUS_INVALID_FEATURE	Not support soft trigger.	
TY_STATUS_IDLE	Camera device is not started	
TY_STATUS_IDLE		
TY_STATUS_IDLE	Suggestions:	
TY_STATUS_IDLE	Suggestions: Please start the camera device first	
TY_STATUS_IDLE	Suggestions: Please start the camera device first Like this:	
TY_STATUS_IDLE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice);	
TY_STATUS_IDLE	Suggestions: Please start the camera device first Like this:	
	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice);	
TY_STATUS_IDLE TY_STATUS_WRONG_MODE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice);	
	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice);	
TY_STATUS_WRONG_MODE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice); Not in trigger mode. Failed to send soft trigger	
TY_STATUS_WRONG_MODE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice); Not in trigger mode. Failed to send soft trigger Suggestions:	
TY_STATUS_WRONG_MODE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice); Not in trigger mode. Failed to send soft trigger Suggestions: Possible reasons:	
TY_STATUS_WRONG_MODE	Suggestions: Please start the camera device first Like this: TYStartCapture(hDevice); TYSendSoftTrigger(hDevice); Not in trigger mode. Failed to send soft trigger Suggestions:	

Return values

TY_STATUS_BUSY	Failed to send soft trigger	
	Suggestions: Possible reasons: 1.Camera is busy, the last soft trigger is not completed,	, please tr

5.1.2.53 TYSetBool()

```
TY_CAPI TYSetBool (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

bool value )
```

Set value of bool feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	value	Bool value.

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYSetBool called with invalid device handle	
	Suggestions: Please check device handle Like this: TYSetBool(hDevice, componentID, featureID, value) ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get corn 2.Memory in stack to store handle data is corrupt 3.After getting handle, you updated device list h	rect handle
TY_STATUS_INVALID_COMPONENT	Invalid component ID	
	Suggestions: Please check componentID parameter Like this: TYSetBool(hDevice, componentID, featureID, value) ^ is invalid componentID should be the value returned by TYGetCo	omponentIDs

TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYSetBool(hDevice, componentID, featureID, value); ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the state of the camera device
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYSetBool(hDevice, componentID, featureID, value); ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY_STATUS_BUSY	Device is capturing, the feature is locked.
TY_STATUS_TIMEOUT	Failed to set bool feature Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the new
TY_STATUS_DEVICE_ERROR	Failed to set bool feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not imple 2.Camera device is abnormal and cannot set bool feature

5.1.2.54 TYSetByteArray()

```
TY_CAPI TYSetByteArray (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

const uint8_t * pBuffer,

uint32_t bufferSize )
```

Write byte array to device.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
out	pBuffer	Byte buffer.
in	bufferSize	Size of buffer.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYSetByteArray called with invalid device handle
	Suggestions: Please check device handle Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the state of
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_NOT_PERMITTED TY_STATUS_WRONG_TYPE	The feature is not writable. Feature type mismatch
	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer: ^ type mismatch
TY_STATUS_WRONG_TYPE	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer: ^ type mismatch The feature type you entered does not match. You can use TYFeature TYSetByteArray called with NULL pointer Suggestions: Please check your code Like this:
TY_STATUS_WRONG_TYPE	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer, ^ type mismatch The feature type you entered does not match. You can use TYFeature TYSetByteArray called with NULL pointer Suggestions: Please check your code Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer,
TY_STATUS_WRONG_TYPE TY_STATUS_NULL_POINTER	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers ^ type mismatch The feature type you entered does not match. You can use TYFeature TYSetByteArray called with NULL pointer Suggestions: Please check your code Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers) is NULL
TY_STATUS_WRONG_TYPE TY_STATUS_NULL_POINTER	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer: ^ type mismatch The feature type you entered does not match. You can use TYFeature TYSetByteArray called with NULL pointer Suggestions: Please check your code Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer: ^ is NULL Array size mismatch Suggestions: Please check the array size Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffer: ^ is in ^ is in
TY_STATUS_WRONG_TYPE TY_STATUS_NULL_POINTER TY_STATUS_WRONG_SIZE	Feature type mismatch Suggestions: Please check the feature type Like this: TYSetByteArray(hDevice, componentID, featureID, pBuffer, buffers

TY_STATUS_DEVICE_ERROR	Failed to set byte array feature	
	Suggestions: Possible reasons: 1.The feature of the camera device is not availal 2.Camera device is abnormal and cannot set byte	-

5.1.2.55 TYSetEnum()

```
TY_CAPI TYSetEnum (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

uint32_t value )
```

Set value of enum feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	value	Enum value.

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	E TYSetEnum called with invalid device handle	
	Suggestions: Please check device handle Like this: TYSetEnum(hDevice, componentID, featureID, value) ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get corr 2.Memory in stack to store handle data is corrupt 3.After getting handle, you updated device list k	rect handle
TY STATUS INVALID COMPONENT	Invalid component ID	, ouring riop
TT_STATOS_INVALID_CONIT CIVENT	'	
	Suggestions: Please check componentID parameter Like this: TYSetEnum(hDevice, componentID, featureID, value)	;
	componentID should be the value returned by TYGetCo You can also view the components of the camera by c	-

Return values

TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYSetEnum(hDevice, componentID, featureID, value); ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetl You can also view the features of the camera device by obtaining the state of the camera device
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYSetEnum(hDevice, componentID, featureID, value); ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY STATUS INVALID PARAMETER	Out of range
	Suggestions: Please check the value Like this: TYSetEnum(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetEnumEntryInfo to get the
TY_STATUS_BUSY	Device is capturing, the feature is locked.
TY_STATUS_TIMEOUT	Failed to set enum feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the ne
TY_STATUS_DEVICE_ERROR	Failed to set enum feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not imple 2.Camera device is abnormal and cannot set enum feature

5.1.2.56 TYSetFloat()

```
TY_CAPI TYSetFloat (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

float value )
```

Set value of float feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	value	Float value.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYSetFloat called with invalid device handle
	Suggestions:
	Please check device handle
	Like this: TYSetFloat(hDevice, componentID, featureID, value);
	^ is invalid
	The hDevice parameter you input is not recorded Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYU
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter Like this:
	TYSetFloat(hDevice, componentID, featureID, value);
	^ is invalid componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions:
	Please check featureID parameter Like this:
	TYSetFloat(hDevice, componentID, featureID, value); ^ is invalid
	You entered an invalid featureID parameter
	You can get a list of features of the camera device through TYGe You can also view the features of the camera device by obtaining
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions:
	Please check the feature type Like this:
	TYSetFloat(hDevice, componentID, featureID, value);
	^ type mismatch The feature type you entered does not match. You can use TYFeatu
TY_STATUS_OUT_OF_RANGE	Out of range
	Suggestions:
	Please check the value Like this:
	LUKE INIS:
	TYSetFloat(hDevice, componentID, featureID, value);
TY_STATUS_BUSY	TYSetFloat(hDevice, componentID, featureID, value);
TY_STATUS_BUSY TY_STATUS_TIMEOUT	TYSetFloat(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the
	TYSetFloat(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the Device is capturing, the feature is locked. Failed to set float feature
	TYSetFloat(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the Device is capturing, the feature is locked. Failed to set float feature Suggestions: Possible reasons:
TY_STATUS_TIMEOUT	TYSetFloat(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the Device is capturing, the feature is locked. Failed to set float feature Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the
	TYSetFloat(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the Device is capturing, the feature is locked. Failed to set float feature Suggestions: Possible reasons:
TY_STATUS_TIMEOUT	TYSetFloat (hDevice, componentID, featureID, value);
TY_STATUS_TIMEOUT	TYSetFloat (hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetFloatRange to get the Device is capturing, the feature is locked. Failed to set float feature Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the Failed to set float feature Suggestions: Possible reasons:
TY_STATUS_TIMEOUT	TYSetFloat(hDevice, componentID, featureID, value);

5.1.2.57 TYSetInt()

```
TY_CAPI TYSetInt (
            TY_DEV_HANDLE hDevice,
             TY_COMPONENT_ID componentID,
             TY_FEATURE_ID featureID,
             int32_t value )
```

Set value of integer feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	value	Integer value.

TV STATUS OV	Succeed.
TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYSetInt called with invalid device handle
	Suggestions:
	Please check device handle
	Like this:
	TYSetInt(hDevice, componentID, featureID, value); ^ is invalid
	The hDevice parameter you input is not recorded
	Possible reasons:
	1.TYOpenDevice failed to open device and get correct handle
	2. Memory in stack to store handle data is corrupted
	3. After getting handle, you updated device list by calling TYU
TY STATUS INVALID COMPONENT	Invalid component ID
	Suggestions:
	Please check componentID parameter
	Like this:
	TYSetInt(hDevice, componentID, featureID, value);
	^ is invalid
	componentID should be the value returned by TYGetComponentIDs
	You can also view the components of the camera by obtaining the
TV STATUS INVALID FEATURE	Invalid feature ID
TY_STATUS_INVALID_FEATURE	invalid leature id
	Suggestions:
	Please check featureID parameter
	Like this:
	TYSetInt(hDevice, componentID, featureID, value);
	^ is invalid
	You entered an invalid featureID parameter

TY_STATUS_NOT_PERMITTED The feature is not writable.

You can get a list of features of the camera device through TYGet You can also view the features of the camera device by obtaining

TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYSetInt(hDevice, componentID, featureID, value); ^ type mismatch The feature type you entered does not match. You can use TYFeatur
TY_STATUS_OUT_OF_RANGE	Out of range
	Suggestions: Please check the value Like this: TYSetInt(hDevice, componentID, featureID, value); ^ is out of range The value is out of range, please use TYGetIntRange to get the ra
TY_STATUS_BUSY	Device is capturing, the feature is locked.
TY_STATUS_TIMEOUT	Failed to set int feature
	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the n
TY_STATUS_DEVICE_ERROR	Failed to set int feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not impl 2.Camera device is abnormal and cannot set int feature

5.1.2.58 TYSetLogLevel()

```
TY_CAPI TYSetLogLevel ( {\tt TY\_LOG\_LEVEL}\ lvl\ )
```

Set log level.

Parameters

in	lvl	Log level.

Return values

```
TY_STATUS_OK Succeed.
```

5.1.2.59 TYSetLogPrefix()

set log prefix

Parameters

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_PARAMETER	Prefix is empty or prefix is too long
	Suggestions: Prefix is empty or prefix is too long, cannot be set Like this: TYSetLogPrefix(prefix); ^ prefix is empty or prefix is too

5.1.2.60 TYSetString()

```
TY_CAPI TYSetString (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,

const char * buffer )
```

Set value of string feature.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	buffer	String buffer.

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYSetString called with invalid device handle	
	Suggestions:	
	Please check device handle	
	Like this:	
	TYSetString(hDevice, componentID, featureID, pBuffe ^ is invalid	er);
	The hDevice parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenDevice failed to open device and get correct	ct handle
	2.Memory in stack to store handle data is corrupted	d
	3.After getting handle, you updated device list by	calling T

TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYSetString(hDevice, componentID, featureID, pBuffer); ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the second
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYSetString(hDevice, componentID, featureID, pBuffer); ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGet You can also view the features of the camera device by obtaining
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYSetString(hDevice, componentID, featureID, pBuffer); ^ type mismatch The feature type you entered does not match. You can use TYFeature
TY_STATUS_NULL_POINTER	TYSetString called with NULL pointer
	Suggestions: Please check your code Like this: TYSetString(hDevice, componentID, featureID, pBuffer);
TY_STATUS_OUT_OF_RANGE	Input string is too long.
TY_STATUS_BUSY	Device is capturing, the feature is locked.
TY_STATUS_TIMEOUT	Failed to set string feature
TY_STATUS_DEVICE_ERROR	Suggestions: Possible reasons: 1.Network communication is abnormal, please check whether the related to set string feature
	Suggestions: Possible reasons: 1.The feature of the camera device is not available or not impleated and cannot set string feature

5.1.2.61 TYSetStruct()

```
TY_CAPI TYSetStruct (

TY_DEV_HANDLE hDevice,

TY_COMPONENT_ID componentID,

TY_FEATURE_ID featureID,
```

void * pStruct, uint32_t structSize)

Set value of struct.

Parameters

in	hDevice	Device handle.
in	componentID	Component ID.
in	featureID	Feature ID.
in	pStruct	Pointer of struct.
in	structSize	Size of struct.

TY_STATUS_OK	Succeed.
TY_STATUS_INVALID_HANDLE	TYSetStruct called with invalid device handle
	Suggestions: Please check device handle Like this: TYSetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get correct handle 2.Memory in stack to store handle data is corrupted 3.After getting handle, you updated device list by calling TYUpo
TY_STATUS_INVALID_COMPONENT	Invalid component ID
	Suggestions: Please check componentID parameter Like this: TYSetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid componentID should be the value returned by TYGetComponentIDs You can also view the components of the camera by obtaining the xr
TY_STATUS_INVALID_FEATURE	Invalid feature ID
	Suggestions: Please check featureID parameter Like this: TYSetStruct(hDevice, componentID, featureID, pStruct, structSize ^ is invalid You entered an invalid featureID parameter You can get a list of features of the camera device through TYGetI You can also view the features of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device by obtaining the camera device of the camera device o
TY_STATUS_NOT_PERMITTED	The feature is not writable.
TY_STATUS_WRONG_TYPE	Feature type mismatch
	Suggestions: Please check the feature type Like this: TYSetStruct(hDevice, componentID, featureID, pStruct, structSize ^ type mismatch The feature type you entered does not match. You can use TYFeature

Return values

TY_STATUS_NULL_POINTER	TYSetStruct called with NULL pointer	
	Suggestions: Please check your code Like this: TYSetStruct(hDevice, componentID, featureID, pSt: ^ is	ruct, structSize
TY_STATUS_WRONG_SIZE	Struct size mismatch	
TY STATUS BUSY	Suggestions: Please check the struct size Like this: TYSetStruct(hDevice, componentID, featureID, pSt: The struct size you entered does not match Device is capturing, the feature is locked.	ruct, structSize ^ is invai
TY_STATUS_TIMEOUT	Failed to set struct feature Suggestions: Possible reasons: 1.Network communication is abnormal, please check	whether the ne
TY_STATUS_DEVICE_ERROR	Failed to set struct feature	
	Suggestions: Possible reasons: 1.The feature of the camera device is not available 2.Camera device is abnormal and cannot set struct	_

5.1.2.62 TYStartCapture()

Start capture.

Parameters

TY_STATUS_OK	Succeed.	
TY_STATUS_INVALID_HANDLE	TYStartCapture called with invalid device handle	
	Suggestions: Please check device handle Like this: TYStartCapture(hDevice); ^ is invalid The hDevice parameter you input is not recorded Possible reasons: 1.TYOpenDevice failed to open device and get corn 2.Memory in stack to store handle data is corrupt 3.After getting handle, you updated device list k	ed

Return values

TY_STATUS_INVALID_COMPONENT	No components are enabled		
	Suggestions: Please enable the components of the camera device: Like this: TYEnableComponents(hDevice, componentIDs); TYStartCapture(hDevice);	īirst	
TY_STATUS_BUSY	Camera device has been started.		
	Suggestions: Please stop the camera device first Like this: TYStopCapture(hDevice); TYStartCapture(hDevice);		
TY_STATUS_DEVICE_ERROR	Failed to start camera		
	Suggestions: Possible reasons: 1.Camera device is abnormal and cannot start the 2.Network communication is abnormal, please check 3.Camera is busy, please try again		n

5.1.2.63 TYStopCapture()

Stop capture.

Parameters

TY_STATUS_OK	Succeed.		
TY_STATUS_INVALID_HANDLE	TYStopCapture called with invalid device handle		
	Suggestions:		
	Please check device handle		
	Like this:		
	TYStopCapture(hDevice);		
	^ is invalid		
	The hDevice parameter you input is not recorded		
	Possible reasons:		
	1.TYOpenDevice failed to open device and get correct hand	le	
	2.Memory in stack to store handle data is corrupted		
	3.After getting handle, you updated device list by callin	g TYUp	

Return values

TY_STATUS_IDLE	Camera device has been stopped	
	Suggestions: The camera device has stopped, usually after starting Like this: TYStartCapture(hDevice); TYStopCapture(hDevice);	
TY_STATUS_DEVICE_ERROR	Failed to stop camera	
	Suggestions: Possible reasons: 1.Camera device is abnormal and cannot stop the camera	

5.1.2.64 TYUpdateAllDeviceList()

Update current connected devices.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.

5.1.2.65 TYUpdateDeviceList()

Update current connected devices.

Parameters

ı			
	in	ifaceHandle	Interface handle.

TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.

Return values

TY_STATUS_INVALID_INTERFACE	TYUpdateDeviceList called with invalid interface handle	
	Suggestions:	
	Please check interface handle	
	Like this:	
	TYUpdateDeviceList(ifaceHandle);	
	^ is invalid	
	The ifaceHandle parameter you input is not recorded	
	Possible reasons:	
	1.TYOpenInterface failed to open interface and get co	orrect handle
	2.Memory in stack to store handle data is corrupted	
	3.After getting handle, you updated interface list by	y calling TYU

5.1.2.66 TYUpdateInterfaceList()

Update current interfaces. call before TYGetInterfaceList.

Return values

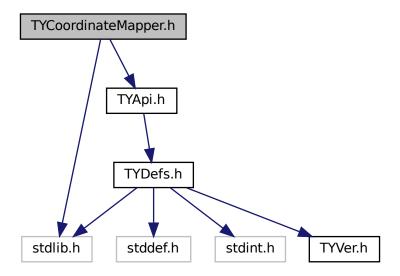
TY_STATUS_OK	Succeed.
TY_STATUS_NOT_INITED	TYInitLib not called.

5.2 TYCoordinateMapper.h File Reference

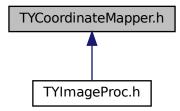
Coordinate Conversion API.

```
#include <stdlib.h>
#include "TYApi.h"
```

Include dependency graph for TYCoordinateMapper.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct TY_PIXEL_DESC
- struct TY_PIXEL_COLOR_DESC

Macros

• #define **TYMAP_CHECKRET**(f, bufToFree)

Typedefs

- typedef struct TY_PIXEL_DESC TY_PIXEL_DESC
- typedef struct TY PIXEL COLOR DESC TY PIXEL COLOR DESC

Functions

TY_CAPI TYInvertExtrinsic (const TY_CAMERA_EXTRINSIC *orgExtrinsic, TY_CAMERA_EXTRINSIC *invExtrinsic)

Calculate 4x4 extrinsic matrix's inverse matrix.

TY_CAPI TYMapDepthToPoint3d (const TY_CAMERA_CALIB_INFO *src_calib, uint32_t depthW, uint32
 _t depthH, const TY_PIXEL_DESC *depthPixels, uint32_t count, TY_VECT_3F *point3d, float f_scale_
 unit=1.0f)

Map pixels on depth image to 3D points.

• TY_CAPI TYMapPoint3dToDepth (const TY_CAMERA_CALIB_INFO *dst_calib, const TY_VECT_3F *point3d, uint32_t count, uint32_t depthW, uint32_t depthH, TY_PIXEL_DESC *depth, float f_scale_← unit=1.0f)

Map 3D points to pixels on depth image. Reverse operation of TYMapDepthToPoint3d.

TY_CAPI TYMapDepthImageToPoint3d (const TY_CAMERA_CALIB_INFO *src_calib, int32_t imageW, int32_t imageH, const uint16_t *depth, TY_VECT_3F *point3d, float f_scale_unit=1.0f)

Map depth image to 3D points. 0 depth pixels maps to (NAN, NAN, NAN).

- TY_CAPI TYDepthImageFillEmptyRegion (uint16_t *depth, uint32_t depthW, uint32_t depthH) Fill depth image empty region.
- TY_CAPI TYMapPoint3dToDepthImage (const TY_CAMERA_CALIB_INFO *dst_calib, const TY_VECT_3F *point3d, uint32_t count, uint32_t depthW, uint32_t depthH, uint16_t *depth, float f_target_scale=1.0f)

Map 3D points to depth image. (NAN, NAN, NAN) will be skipped.

TY_CAPI TYMapPoint3dToPoint3d (const TY_CAMERA_EXTRINSIC *extrinsic, const TY_VECT_3F *point3dFrom, int32 t count, TY VECT 3F *point3dTo)

Map 3D points to another coordinate.

5.2.1 Detailed Description

Coordinate Conversion API.

Note

Considering performance, we leave the responsibility of parameters check to users.

Copyright

Copyright(C)2016-2018 Percipio All Rights Reserved

5.2.2 Macro Definition Documentation

5.2.2.1 TYMAP_CHECKRET

Definition at line 274 of file TYCoordinateMapper.h.

5.2.3 Function Documentation

5.2.3.1 TYDepthImageFillEmptyRegion()

Fill depth image empty region.

Parameters

in	n depth Depth image pixels.	
in depthW Width of current		Width of current depth image.
in	depthH	Height of current depth image.

5.2.3.2 TYInvertExtrinsic()

Calculate 4x4 extrinsic matrix's inverse matrix.

Parameters

in <i>orgExtrinsic</i>		Input extrinsic matrix.	
out	invExtrinsic	Inverse matrix.	

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_ERROR	Calculation failed.

5.2.3.3 TYMapDepthImageToPoint3d()

Map depth image to 3D points. 0 depth pixels maps to (NAN, NAN, NAN).

Parameters

in	src_calib	Depth image's calibration data.
in	depthW	Width of depth image.
in	depthH	Height of depth image.
in	depth	Depth image.
out	point3d	Output point3D image.

Return values

TY_STATUS_OK	Succeed.
--------------	----------

5.2.3.4 TYMapDepthToPoint3d()

Map pixels on depth image to 3D points.

Parameters

in	src_calib	Depth image's calibration data.
in	depthW	Width of depth image.

Parameters

in	depthH	Height of depth image.
in	depthPixels	Pixels on depth image.
in	count	Number of depth pixels.
out	point3d	Output point3D.

Return values

TY STATUS OK	Succeed.
--------------	----------

5.2.3.5 TYMapPoint3dToDepth()

Map 3D points to pixels on depth image. Reverse operation of TYMapDepthToPoint3d.

Parameters

in	dst_calib	Target depth image's calibration data.	
in	point3d	Input 3D points.	
in	count Number of points.		
in	depthW	Width of target depth image.	
in	in depthH Height of target depth image.		
out	depth	Output depth pixels.	

Return values

```
TY_STATUS_OK Succeed.
```

5.2.3.6 TYMapPoint3dToDepthImage()

```
uint32_t depthH,
uint16_t * depth,
float f_target_scale = 1.0f )
```

Map 3D points to depth image. (NAN, NAN, NAN) will be skipped.

Parameters

in	dst_calib	Target depth image's calibration data.
in	point3d	Input 3D points.
in	count	Number of points.
in <i>depthW</i>		Width of target depth image.
in <i>depthH</i>		Height of target depth image.
in,out	depth	Depth image buffer.

Return values

TY_STATUS_OK	Succeed.
--------------	----------

5.2.3.7 TYMapPoint3dToPoint3d()

Map 3D points to another coordinate.

Parameters

in	extrinsic	Extrinsic matrix.
in	point3dFrom	Source 3D points.
in	count	Number of source 3D points.
out	point3dTo	Target 3D points.

Return values

```
TY_STATUS_OK Succeed.
```

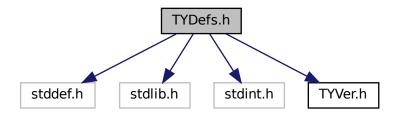
5.3 TYDefs.h File Reference

TYDefs.h includes camera control and data receiving data definitions which supports configuration for image resolution, frame rate, exposure time, gain, working mode,etc.

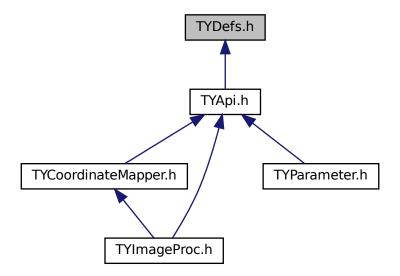
5.3 TYDefs.h File Reference

```
#include <stddef.h>
#include <stdlib.h>
#include <stdint.h>
#include "TYVer.h"
```

Include dependency graph for TYDefs.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct TY_VERSION_INFO
- struct TY_DEVICE_NET_INFO

device network information

- struct TY_DEVICE_USB_INFO
- struct TY_INTERFACE_INFO
- struct TY_DEVICE_BASE_INFO

- struct TY_FEATURE_INFO
- struct TY_INT_RANGE
- struct TY_FLOAT_RANGE

float range data structure

struct TY BYTEARRAY ATTR

byte array data structure

- struct TY ENUM ENTRY
- struct TY_VECT_3F
- struct TY CAMERA INTRINSIC
- struct TY_CAMERA_EXTRINSIC
- struct TY_CAMERA_DISTORTION
- struct TY_CAMERA_CALIB_INFO
- struct TY_TRIGGER_PARAM
- struct TY TRIGGER PARAM EX
- struct TY_TRIGGER_TIMER_LIST
- struct TY_TRIGGER_TIMER_PERIOD
- struct TY_AEC_ROI_PARAM
- struct TY_PHC_GROUP_ATTR
- · struct TY PHC GROUP ATTR::phc group attr
- struct pattern_sine_param
- struct pattern_gray_param
- struct pattern_bin_param
- struct TY_LASER_PATTERN_PARAM
- struct TY CAMERA STATISTICS
- struct TY_IMU_DATA
- struct TY_ACC_BIAS
- struct TY_ACC_MISALIGNMENT
- struct TY_ACC_SCALE
- struct TY_GYRO_BIAS
- struct TY_GYRO_MISALIGNMENT
- struct TY_GYRO_SCALE
- struct TY_CAMERA_TO_IMU
- struct TY_TOF_FREQ
- struct TY LASER PARAM
- struct TY IMAGE DATA
- struct TY_FRAME_DATA
- struct TY EVENT INFO
- struct TY_DO_WORKMODE
- struct TY DI WORKMODE
- struct TY_TEMP_DATA
- struct TY_CAMERA_ROTATION

Macros

- #define STDBOOL H
- #define __bool_true_false_are_defined 1
- #define bool _Bool
- #define true 1
- #define false 0
- #define TY_DLLIMPORT __attribute__((visibility("default")))
- #define TY_DLLEXPORT __attribute__((visibility("default")))
- · #define TY STDC
- #define TY_CDEC

- #define TY_EXPORT TY_DLLIMPORT
- #define TY_EXTC
- #define TY_CAPI TY_EXTC TY_EXPORT TY_STATUS TY_STDC
- · #define TY_INT_SGBM_COST_PARAMITY_INT_SGBM_UNIQUE_MAX_COST
- #define TY_BOOL_FLASHLIGHT TY_BOOL_IR_FLASHLIGHT
- #define TY_INT_FLASHLIGHT_INTENSITY TY_INT_IR_FLASHLIGHT_INTENSITY
- #define TY_INT_AE_TARGET_V TY_INT_AE_TARGET_Y
- #define TY DECLARE IMAGE MODE1(pix)

Typedefs

typedef enum TY STATUS LIST TY STATUS LIST

API call return status.

- typedef int32 t TY_STATUS
- typedef enum TY_FW_ERRORCODE_LIST TY_FW_ERRORCODE_LIST
- typedef uint32_t TY_FW_ERRORCODE
- typedef enum TY EVENT LIST TY ENENT LIST
- typedef int32 t TY EVENT
- typedef void * TY INTERFACE HANDLE

Interface handle.

typedef void * TY DEV HANDLE

Device Handle.

- typedef enum TY_DEVICE_COMPONENT_LIST TY_DEVICE_COMPONENT_LIST
- typedef uint32_t TY_COMPONENT_ID

component unique id

• typedef enum TY_FEATURE_TYPE_LIST TY_FEATURE_TYPE_LIST

Feature Format Type definitions.

- typedef uint32_t TY_FEATURE_TYPE
- typedef enum TY FEATURE ID LIST TY FEATURE ID LIST

feature for component definitions

typedef uint32_t TY_FEATURE_ID

feature unique id

- typedef enum TY CONFIG MODE LIST TY CONFIG MODE LIST
- typedef uint32 t TY CONFIG MODE
- typedef enum TY_DEPTH_QUALITY_LIST TY_DEPTH_QUALITY_LIST
- typedef uint32_t TY_DEPTH_QUALITY
- typedef enum TY_TRIGGER_POL_LIST TY_TRIGGER_POL_LIST

set external trigger signal edge

- typedef uint32_t TY_TRIGGER_POL
- typedef enum TY_INTERFACE_TYPE_LIST TY_INTERFACE_TYPE_LIST
- typedef uint32 t TY_INTERFACE_TYPE
- typedef enum TY_ACCESS_MODE_LIST TY_ACCESS_MODE_LIST
- typedef uint8_t TY_ACCESS_MODE
- typedef enum TY_STREAM_ASYNC_MODE_LIST TY_STREAM_ASYNC_MODE_LIST

stream async mode

- typedef uint8_t TY_STREAM_ASYNC_MODE
- typedef enum TYPixFmtList TYPixFmtList

pixel format definitions

- typedef uint32 t TYPixFmt
- typedef enum TY_PIXEL_BITS_LIST TY_PIXEL_BITS_LIST

- typedef uint32_t TY_PIXEL_BITS
- typedef enum TY_PIXEL_FORMAT_LIST TY_PIXEL_FORMAT_LIST

pixel format definitions

- typedef uint32 t TY PIXEL FORMAT
- typedef enum TY RESOLUTION MODE LIST TY RESOLUTION MODE LIST

predefined resolution list

- typedef int32_t TY_RESOLUTION_MODE
- typedef enum TY IMAGE MODE LIST TY IMAGE MODE LIST

Predefined Image Mode List image mode controls image resolution & format predefined image modes named like TY_IMAGE_MODE_MONO_160x120,TY_IMAGE_MODE_RGB_1280x960.

- typedef uint32_t TY_IMAGE_MODE
- typedef enum TY_TRIGGER_MODE_LIST TY_TRIGGER_MODE_LIST
- typedef int16_t TY_TRIGGER_MODE
- typedef enum TY_TIME_SYNC_TYPE_LIST TY_TIME_SYNC_TYPE_LIST

type of time sync

- typedef uint32_t TY_TIME_SYNC_TYPE
- typedef uint32_t TY_E_VOLT_T
- typedef uint32_t TY_E_DO_MODE
- typedef uint32 t TY E DI MODE
- typedef uint32 t TY E DI INT ACTION
- typedef uint32_t TY_TEMPERATURE_ID
- typedef enum TY LOG LEVEL LIST TY LOG LEVEL LIST
- typedef int32 t TY LOG LEVEL
- typedef struct TY_VERSION_INFO TY_VERSION_INFO
- typedef struct TY_DEVICE_NET_INFO TY_DEVICE_NET_INFO

device network information

- typedef struct TY_DEVICE_USB_INFO TY_DEVICE_USB_INFO
- typedef struct TY_INTERFACE_INFO TY_INTERFACE_INFO
- typedef struct TY DEVICE BASE INFO TY DEVICE BASE INFO
- typedef enum TY_VISIBILITY_TYPE TY_VISIBILITY_TYPE
- typedef struct TY_FEATURE_INFO TY_FEATURE_INFO
- typedef struct TY_INT_RANGE TY_INT_RANGE
- typedef struct TY_FLOAT_RANGE TY_FLOAT_RANGE

float range data structure

typedef struct TY_BYTEARRAY_ATTR TY_BYTEARRAY_ATTR

byte array data structure

- typedef struct TY_ENUM_ENTRY TY_ENUM_ENTRY
- typedef struct TY_VECT_3F TY_VECT_3F
- typedef struct TY CAMERA INTRINSIC TY CAMERA INTRINSIC
- typedef struct TY CAMERA EXTRINSIC TY CAMERA EXTRINSIC
- typedef struct TY_CAMERA_DISTORTION TY_CAMERA_DISTORTION
- typedef struct TY_CAMERA_CALIB_INFO TY_CAMERA_CALIB_INFO
- typedef struct TY_TRIGGER_PARAM TY_TRIGGER_PARAM
- typedef struct TY_TRIGGER_PARAM_EX TY_TRIGGER_PARAM_EX
- typedef struct TY_TRIGGER_TIMER_LIST TY_TRIGGER_TIMER_LIST
- typedef struct TY TRIGGER TIMER PERIOD TY TRIGGER TIMER PERIOD
- typedef struct TY_AEC_ROI_PARAM TY_AEC_ROI_PARAM
- typedef struct TY_PHC_GROUP_ATTR TY_PHC_GROUP_ATTR
- typedef struct TY_LASER_PATTERN_PARAM TY_LASER_PATTERN_PARAM
- typedef struct TY_CAMERA_STATISTICS TY_CAMERA_STATISTICS
- typedef struct TY_IMU_DATA TY_IMU_DATA
- typedef struct TY_ACC_BIAS TY_ACC_BIAS
- typedef struct TY ACC MISALIGNMENT TY ACC MISALIGNMENT
- typedef struct TY_ACC_SCALE TY_ACC_SCALE

- typedef struct TY_GYRO_BIAS TY_GYRO_BIAS
- typedef struct TY_GYRO_MISALIGNMENT TY_GYRO_MISALIGNMENT
- typedef struct TY_GYRO_SCALE TY_GYRO_SCALE
- typedef struct TY CAMERA TO IMU TY CAMERA TO IMU
- typedef struct TY TOF FREQ TY TOF FREQ
- typedef enum TY_IMU_FPS_LIST TY_IMU_FPS_LIST
- typedef struct TY LASER PARAM TY_LASER_PARAM
- typedef struct TY IMAGE DATA TY IMAGE DATA
- typedef struct TY FRAME DATA TY FRAME DATA
- typedef struct TY EVENT INFO TY EVENT INFO
- typedef struct TY DO WORKMODE TY DO WORKMODE
- typedef struct TY DI WORKMODE TY DI WORKMODE
- typedef struct TY_TEMP_DATA TY_TEMP_DATA
- typedef struct TY_CAMERA_ROTATION TY_CAMERA_ROTATION

= 0x00040000, TY COMPONENT IR CAM RIGHT = 0x00080000,

TY_COMPONENT_RGB_CAM = TY_COMPONENT_RGB_CAM_LEFT }

TY COMPONENT LASER = 0x00400000, TY COMPONENT IMU = 0x00800000,

Enumerations

```
    enum TY STATUS LIST: int32 t {

 TY STATUS OK = 0, TY STATUS ERROR = -1001, TY STATUS NOT INITED = -1002, TY STATUS ↔
 _NOT_IMPLEMENTED = -1003,
 TY_STATUS_NOT_PERMITTED = -1004, TY_STATUS_DEVICE_ERROR = -1005, TY_STATUS_INVA
 LID_PARAMETER = -1006, TY_STATUS_INVALID_HANDLE = -1007,
 TY STATUS INVALID COMPONENT = -1008, TY STATUS INVALID FEATURE = -1009, TY STATU
 S_WRONG_TYPE = -1010, TY_STATUS_WRONG_SIZE = -1011,
 TY_STATUS_OUT_OF_MEMORY = -1012, TY_STATUS_OUT_OF_RANGE = -1013, TY_STATUS_TIM ←
 EOUT = -1014, TY STATUS WRONG MODE = -1015,
 TY STATUS BUSY = -1016, TY STATUS IDLE = -1017, TY STATUS NO DATA = -1018, TY STATU↔
 S NO BUFFER = -1019,
 TY STATUS NULL POINTER = -1020, TY STATUS READONLY FEATURE = -1021, TY STATUS I↔
 NVALID_DESCRIPTOR = -1022, TY_STATUS_INVALID_INTERFACE = -1023,
 TY STATUS_FIRMWARE_ERROR = -1024, TY_STATUS_ERROR_XML = -1025, TY_STATUS_DEV_E ←
 PERM = -1, TY_STATUS_DEV_EIO = -5,
 TY STATUS DEV ENOMEM = -12, TY STATUS DEV EBUSY = -16, TY STATUS DEV EINVAL = -22
 }
    API call return status.
enum TY FW ERRORCODE LIST : uint32 t {
 TY FW ERRORCODE CAMO NOT DETECTED = 0x00000001, TY FW ERRORCODE CAM1 NOT ←
 DETECTED = 0x00000002, TY_FW_ERRORCODE_CAM2_NOT_DETECTED = 0x00000004, TY_FW_E ←
 RRORCODE_POE_NOT_INIT = 0x00000008,
 TY FW ERRORCODE RECMAP NOT CORRECT = 0x00000010, TY FW ERRORCODE LOOKUPT ←
 ABLE NOT CORRECT = 0x00000020. TY FW ERRORCODE DRV8899 NOT INIT = 0x00000040. T↔
 Y FW ERRORCODE FOC START ERR = 0 \times 000000080,
 TY FW ERRORCODE CONFIG NOT FOUND = 0x00010000, TY FW ERRORCODE CONFIG NOT⊷
 _CORRECT = 0x00020000, TY_FW_ERRORCODE_XML_NOT_FOUND = 0x00040000, TY_FW_ERRO↔
 RCODE XML NOT CORRECT = 0 \times 00080000.
 TY FW ERRORCODE XML OVERRIDE FAILED = 0x00100000, TY FW ERRORCODE CAM INIT ←
 FAILED = 0x00200000, TY FW ERRORCODE LASER INIT FAILED = 0x00400000 }

    enum TY_EVENT_LIST: int32_t { TY_EVENT_DEVICE_OFFLINE = -2001, TY_EVENT_LICENSE_ERR ←

 OR = -2002, TY_EVENT_FW_INIT_ERROR = -2003 }

    enum TY DEVICE COMPONENT LIST: uint32 t {
```

TY_COMPONENT_DEVICE = 0x80000000, TY_COMPONENT_DEPTH_CAM = 0x00010000, TY_COMPONENT_IR_CAM_LI

TY COMPONENT RGB CAM LEFT = 0x00100000, TY COMPONENT RGB CAM RIGHT = 0x00200000,

TY COMPONENT BRIGHT HISTO = 0x01000000, TY COMPONENT STORAGE = 0x02000000,

```
• enum TY_FEATURE_TYPE_LIST : uint32_t {
 TY FEATURE CMD = 0x0000, TY FEATURE INT = 0x1000, TY FEATURE FLOAT = 0X2000, TY FE
 ATURE ENUM = 0 \times 3000.
 TY_FEATURE_BOOL = 0x4000, TY_FEATURE_STRING = 0x5000, TY_FEATURE_BYTEARRAY =
 0x6000, TY_FEATURE_STRUCT = 0x7000 }
    Feature Format Type definitions.
• enum TY FEATURE ID LIST: uint32 t {
 TY_STRUCT_CAM_INTRINSIC = 0x0000 | TY_FEATURE_STRUCT, TY_STRUCT_EXTRINSIC_TO_DEPTH
 = 0x0001 | TY FEATURE STRUCT, TY STRUCT EXTRINSIC TO IR LEFT = 0x0002 | TY FEATURE ↔
 STRUCT, TY STRUCT CAM RECTIFIED ROTATION = 0x0003 | TY FEATURE STRUCT.
 TY_STRUCT_CAM_DISTORTION = 0x0006 | TY_FEATURE_STRUCT, TY_STRUCT_CAM_CALIB_DATA
 = 0x0007 | TY FEATURE STRUCT, TY STRUCT CAM RECTIFIED INTRI = 0x0008 | TY FEATURE ←
 STRUCT, TY BYTEARRAY CUSTOM BLOCK = 0x000A | TY FEATURE BYTEARRAY,
 TY_BYTEARRAY_ISP_BLOCK = 0x000B | TY_FEATURE_BYTEARRAY, TY_INT_PERSISTENT_IP =
 0x0010 | TY_FEATURE_INT, TY_INT_PERSISTENT_SUBMASK = 0x0011 | TY_FEATURE_INT, TY_IN↔
 T_PERSISTENT_GATEWAY = 0x0012 | TY FEATURE INT,
 FEATURE_INT, TY_INT_ACCEPTABLE_PERCENT = 0x0015 | TY_FEATURE_INT, TY_INT_NTP_SERVER_IP
 = 0x0016 | TY FEATURE INT,
 TY INT PACKET SIZE = 0x0017 | TY FEATURE INT, TY INT LINK CMD TIMEOUT = 0x0018 | TY F↔
 EATURE INT, TY STRUCT CAM STATISTICS = 0x00ff | TY FEATURE STRUCT, TY INT WIDTH MAX
 = 0x0100 | TY FEATURE INT.
 TY INT HEIGHT MAX = 0x0101 | TY FEATURE INT, TY INT OFFSET X = 0x0102 | TY FEATURE INT,
 TY INT OFFSET Y = 0x0103 | TY FEATURE INT, TY INT WIDTH = 0x0104 | TY FEATURE INT,
 TY INT HEIGHT = 0x0105 | TY FEATURE INT, TY ENUM IMAGE MODE = 0x0109 | TY FEATURE ←
 ENUM, TY FLOAT SCALE UNIT = 0x010a | TY FEATURE FLOAT, TY ENUM TRIGGER POL = 0x0201
 TY_FEATURE_ENUM,
 TY_INT_FRAME_PER_TRIGGER = 0x0202 | TY_FEATURE_INT, TY_ENUM_USER_SET_DEFAULT =
 0x0301 | TY FEATURE ENUM, TY ENUM USER SET CURRENT = 0x0304 | TY FEATURE ENUM,
 TY ENUM USER SET SELECTOR = 0x0302 | TY FEATURE ENUM,
 TY CMD USER SET LOAD = 0x0305 | TY FEATURE CMD, TY CMD USER SET SAVE = 0x0303
 TY FEATURE CMD, TY STRING USER SET DESCRIPTION = 0x0306 | TY FEATURE STRING,
 TY STRUCT TRIGGER PARAM = 0x0523 | TY FEATURE STRUCT,
 TY STRUCT TRIGGER PARAM EX = 0x0525 | TY FEATURE STRUCT, TY STRUCT TRIGGER TIMER LIST
 = 0x0526 | TY_FEATURE_STRUCT, TY_STRUCT_TRIGGER_TIMER_PERIOD = 0x0527 | TY_FEATUR↔
 E STRUCT, TY BOOL KEEP ALIVE ONOFF = 0x0203 | TY FEATURE BOOL,
 TY INT KEEP ALIVE TIMEOUT = 0x0204 | TY FEATURE INT, TY BOOL CMOS SYNC = 0x0205 | T↔
 Y_FEATURE_BOOL, TY_INT_TRIGGER_DELAY_US = 0x0206 | TY_FEATURE_INT, TY_BOOL_TRIGGER_OUT_IO
 = 0x0207 | TY FEATURE BOOL,
 TY INT TRIGGER DURATION US = 0x0208 | TY FEATURE INT, TY ENUM STREAM ASYNC
 = 0x0209 | TY FEATURE ENUM, TY INT CAPTURE TIME US = 0x0210 | TY FEATURE INT.
 TY ENUM TIME SYNC TYPE = 0x0211 | TY FEATURE ENUM,
 TY BOOL TIME SYNC READY = 0x0212 | TY FEATURE BOOL, TY BOOL IR FLASHLIGHT =
 0x0213 | TY FEATURE BOOL, TY INT IR FLASHLIGHT INTENSITY = 0x0214 | TY FEATURE INT,
 TY_STRUCT_DO0_WORKMODE = 0x0215 | TY_FEATURE_STRUCT,
 TY_STRUCT_DIO_WORKMODE = 0x0216 | TY_FEATURE_STRUCT, TY_STRUCT_DO1_WORKMODE =
 0x0217 | TY_FEATURE_STRUCT, TY_STRUCT_DI1_WORKMODE = 0x0218 | TY_FEATURE_STRUCT,
 TY STRUCT DO2 WORKMODE = 0x0219 | TY FEATURE STRUCT,
 TY_STRUCT_DI2_WORKMODE = 0x0220 | TY_FEATURE_STRUCT, TY_BOOL_RGB_FLASHLIGHT =
 0x0221 | TY_FEATURE_BOOL, TY_INT_RGB_FLASHLIGHT_INTENSITY = 0x0222 | TY_FEATURE_INT,
 TY ENUM CONFIG MODE = 0x0221 | TY FEATURE ENUM,
 TY ENUM TEMPERATURE ID = 0x0223 | TY FEATURE ENUM, TY STRUCT TEMPERATURE =
 0x0224 | TY FEATURE STRUCT, TY BOOL AUTO EXPOSURE = 0x0300 | TY FEATURE BOOL,
 TY INT EXPOSURE TIME = 0x0301 | TY FEATURE INT,
 TY BOOL AUTO GAIN = 0x0302 | TY FEATURE BOOL, TY INT GAIN = 0x0303 | TY FEATURE INT,
 TY BOOL AUTO AWB = 0x0304 | TY FEATURE BOOL, TY STRUCT AEC ROI = 0x0305 | TY FEA↔
 TURE STRUCT,
```

TY INT TOF HDR RATIO = 0x0306 | TY FEATURE INT, TY INT TOF JITTER THRESHOLD =

```
0x0307 | TY FEATURE INT, TY FLOAT EXPOSURE TIME US = 0x0308 | TY FEATURE FLOAT,
TY INT LASER POWER = 0x0500 | TY FEATURE INT,
TY BOOL LASER AUTO CTRL = 0x0501 | TY FEATURE BOOL, TY STRUCT LASER PATTERN =
0x0502 | TY_FEATURE_STRUCT, TY_INT_LASER_CAM_TRIG_POS = 0x0503 | TY_FEATURE_INT, T↔
Y INT LASER CAM TRIG LEN = 0x0504 | TY FEATURE INT,
TY INT LASER LUT TRIG POS = 0x0505 | TY FEATURE INT, TY INT LASER LUT NUM = 0x0506 |
TY FEATURE INT, TY INT LASER PATTERN OFFSET = 0x0507 | TY FEATURE INT, TY INT LAS↔
ER_MIRROR_NUM = 0x0508 | TY_FEATURE_INT,
TY INT LASER MIRROR SEL = 0x0509 | TY FEATURE INT, TY INT LASER LUT IDX = 0x050a |
TY FEATURE INT, TY INT LASER FACET IDX = 0x050b | TY FEATURE INT, TY INT LASER FA ↔
CET_POS = 0x050c | TY FEATURE INT,
TY_INT_LASER_MODE = 0x050d | TY_FEATURE_INT, TY_INT_CONST_DRV_DUTY = 0x050e
TY FEATURE INT, TY STRUCT LASER ENABLE BY IDX = 0x0530 | TY FEATURE STRUCT,
TY STRUCT LASER POWER BY IDX = 0x0531 | TY FEATURE STRUCT,
TY_STRUCT_FLOOD_ENABLE_BY_IDX = 0x0532 | TY_FEATURE_STRUCT, TY_STRUCT_FLOOD_POWER_BY_IDX
= 0x0533 | TY_FEATURE_STRUCT, TY_BOOL_UNDISTORTION = 0x0510 | TY_FEATURE_BOOL,
TY BOOL BRIGHTNESS HISTOGRAM = 0x0511 | TY FEATURE BOOL,
TY BOOL DEPTH POSTPROC = 0x0512 | TY FEATURE BOOL, TY INT R GAIN = 0x0520 | TY FE↔
ATURE INT, TY INT G GAIN = 0x0521 | TY FEATURE INT, TY INT B GAIN = 0x0522 | TY FEATUR ←
E_INT.
TY INT ANALOG GAIN = 0x0524 | TY FEATURE INT, TY BOOL HDR = 0x0525 | TY FEATURE BOOL,
TY BYTEARRAY HDR PARAMETER = 0x0526 | TY FEATURE BYTEARRAY, TY INT AE TARGET Y
= 0x0527 | TY FEATURE INT,
TY BOOL IMU DATA ONOFF = 0x0600 | TY FEATURE BOOL, TY STRUCT IMU ACC BIAS = 0x0601
TY FEATURE STRUCT, TY STRUCT IMU ACC MISALIGNMENT = 0x0602 TY FEATURE STRUCT,
TY STRUCT IMU ACC SCALE = 0x0603 | TY FEATURE STRUCT,
TY_STRUCT_IMU_GYRO_BIAS = 0x0604 | TY_FEATURE_STRUCT, TY_STRUCT_IMU_GYRO_MISALIGNMENT
= 0x0605 | TY FEATURE STRUCT, TY STRUCT IMU GYRO SCALE = 0x0606 | TY FEATURE STR↔
UCT, TY STRUCT IMU CAM TO IMU = 0x0607 | TY FEATURE STRUCT,
TY ENUM IMU FPS = 0x0608 | TY FEATURE ENUM, TY INT SGBM IMAGE NUM = 0x0610 | TY F↔
EATURE_INT, TY_INT_SGBM_DISPARITY_NUM = 0x0611 | TY_FEATURE_INT, TY_INT_SGBM_DISPARITY_OFFSET
= 0x0612 | TY FEATURE INT,
TY INT SGBM MATCH WIN HEIGHT = 0x0613 | TY FEATURE INT, TY INT SGBM SEMI PARAM P1
= 0x0614 | TY_FEATURE_INT, TY_INT_SGBM_SEMI_PARAM_P2 = 0x0615 | TY_FEATURE_INT,
TY_INT_SGBM_UNIQUE_FACTOR = 0x0616 | TY_FEATURE_INT,
TY_INT_SGBM_UNIQUE_ABSDIFF = 0x0617 | TY_FEATURE_INT, TY_INT_SGBM_UNIQUE_MAX_COST
= 0x0618 | TY FEATURE INT, TY BOOL SGBM HFILTER HALF WIN = 0x0619 | TY FEATURE BOOL,
TY_INT_SGBM_MATCH_WIN_WIDTH = 0x061A | TY_FEATURE_INT,
TY BOOL SGBM MEDFILTER = 0x061B | TY FEATURE BOOL, TY BOOL SGBM LRC = 0x061C | T↔
Y FEATURE BOOL, TY INT SGBM LRC DIFF = 0x061D | TY FEATURE INT, TY INT SGBM MEDFILTER THRESH
= 0x061E | TY FEATURE INT.
TY INT SGBM SEMI PARAM P1 SCALE = 0x061F | TY FEATURE INT, TY INT SGPM PHASE NUM
= 0x0620 | TY FEATURE INT, TY INT SGPM NORMAL PHASE SCALE = 0x0621 | TY FEATURE INT,
TY INT SGPM NORMAL PHASE OFFSET = 0x0622 | TY FEATURE INT,
TY INT SGPM REF PHASE SCALE = 0x0623 | TY FEATURE INT, TY INT SGPM REF PHASE OFFSET
= 0x0624 | TY_FEATURE_INT, TY_FLOAT_SGPM_EPI_HS = 0x0625 | TY_FEATURE_FLOAT,
TY INT SGPM EPI HF = 0x0626 | TY FEATURE INT,
TY BOOL SGPM EPI EN = 0x0627 | TY FEATURE BOOL, TY INT SGPM EPI CH0 = 0x0628 | TY↔
FEATURE INT, TY INT SGPM EPI CH1 = 0x0629 | TY FEATURE INT, TY INT SGPM EPI THRESH
= 0x062A | TY FEATURE INT,
TY BOOL SGPM ORDER FILTER EN = 0x062B | TY FEATURE BOOL, TY INT SGPM ORDER FILTER CHN
= 0x062C | TY FEATURE INT, TY INT DEPTH MIN MM = 0x062D | TY FEATURE INT, TY INT DEPTH MAX MM
= 0x062E | TY FEATURE INT,
TY_INT_SGBM_TEXTURE_OFFSET = 0x062F | TY_FEATURE_INT, TY_INT_SGBM_TEXTURE_THRESH
= 0x0630 | TY_FEATURE_INT, TY_STRUCT_PHC_GROUP_ATTR = 0x0710 | TY_FEATURE_STRUCT,
TY_ENUM_DEPTH_QUALITY = 0x0900 | TY_FEATURE_ENUM,
TY_INT_FILTER_THRESHOLD = 0x0901 | TY_FEATURE_INT, TY_INT_TOF_CHANNEL = 0x0902
TY_FEATURE_INT, TY_INT_TOF_MODULATION_THRESHOLD = 0x0903 | TY_FEATURE_INT,
```

```
TY STRUCT TOF FREQ = 0x0904 | TY FEATURE STRUCT,
 TY BOOL TOF ANTI INTERFERENCE = 0x0905 | TY FEATURE BOOL, TY INT TOF ANTI SUNLIGHT INDEX
 = 0x0906 | TY_FEATURE_INT, TY_INT_MAX_SPECKLE_SIZE = 0x0907 | TY_FEATURE_INT,
 TY_INT_MAX_SPECKLE_DIFF = 0x0908 | TY_FEATURE_INT }
    feature for component definitions
• enum TY CONFIG MODE LIST: uint32 t {
 TY CONFIG MODE PRESETO = 0, TY CONFIG MODE PRESET1, TY CONFIG MODE PRESET2,
 TY CONFIG MODE USERSET0 = (1 << 16),
 TY_CONFIG_MODE_USERSET1, TY_CONFIG_MODE_USERSET2 }

    enum TY_DEPTH_QUALITY_LIST: uint32_t { TY_DEPTH_QUALITY_BASIC = 1, TY_DEPTH_QUALIT

 Y_MEDIUM = 2, TY_DEPTH_QUALITY_HIGH = 4 }

    enum TY TRIGGER POL LIST: uint32 t { TY TRIGGER POL FALLINGEDGE = 0, TY TRIGGER P←

 OL RISINGEDGE = 1 }
    set external trigger signal edge

    enum TY INTERFACE TYPE LIST: uint32 t {

 TY_INTERFACE_UNKNOWN = 0, TY_INTERFACE_RAW = 1, TY_INTERFACE_USB = 2, TY_INTERF ←
 ACE\_ETHERNET = 4,
 TY_INTERFACE_IEEE80211 = 8, TY_INTERFACE_ALL = 0xffff }

    enum TY ACCESS MODE LIST: uint32 t { TY ACCESS READABLE = 0x1, TY ACCESS WRITABLE

 = 0x2

    enum TY STREAM ASYNC MODE LIST: uint32 t {

 TY STREAM ASYNC OFF = 0, TY STREAM ASYNC DEPTH = 1, TY STREAM ASYNC RGB = 2, T↔
 Y STREAM ASYNC DEPTH RGB = 3,
 TY_STREAM_ASYNC_ALL = 0xff }
    stream async mode
enum TYPixFmtList : uint32 t {
 TYPixelFormatMono8 = 0x01080001, TYPixelFormatBayerGBRG8 = 0x0108000A, TYPixelFormat↔
 BayerBGGR8 = 0x0108000B, TYPixelFormatBayerGRBG8 = 0x01080008,
 TYPixelFormatBayerRGGB8 = 0x01080009, TYPixelFormatPacketMono10 = 0x010A0046, TYPixel←
 FormatPacketBayerGBRG10 = 0x010A0054, TYPixelFormatPacketBayerBGGR10 = 0x010A0052,
 TYPixelFormatPacketBayerGRBG10 = 0x010A0056, TYPixelFormatPacketBayerRGGB10 = 0x010←
 A0058, TYPixelFormatPacketMono12 = 0x010C0047, TYPixelFormatPacketBayerGBRG12 = 0x010←
 C0055.
 TYPixelFormatPacketBayerBGGR12 = 0x010C0053. TYPixelFormatPacketBayerGRBG12 = 0x010←
 C0057, TYPixelFormatPacketBayerRGGB12 = 0x010C0059, TYPixelFormatMono10 = 0x810A0046,
 TYPixelFormatBayerGBRG10 = 0x810A0054, TYPixelFormatBayerBGGR10 = 0x810A0052, TYPixel←
 FormatBayerGRBG10 = 0x810A0056, TYPixelFormatBayerRGGB10 = 0x810A0058,
 TYPixelFormatMono12 = 0x810C0047, TYPixelFormatBayerGBRG12 = 0x810C0055, TYPixelFormat⊷
 BayerBGGR12 = 0x810C0053, TYPixelFormatBayerGRBG12 = 0x810C0057,
 TYPixelFormatBayerRGGB12 = 0x810C0059, TYPixelFormatMono14 = 0x810E0104, TYPixelFormat ←
 BayerGBRG14 = 0x810E0107, TYPixelFormatBayerBGGR14 = 0x810E0108,
 TYPixelFormatBayerGRBG14 = 0x810E0105, TYPixelFormatBayerRGGB14 = 0x810E0106, TYPixel←
 \textbf{FormatMono16} = 0x01100007, \textbf{TYPixelFormatBayerGBRG16} = 0x01100030,
 TYPixelFormatBayerBGR16 = 0x01100031, TYPixelFormatBayerGRBG16 = 0x0110002E, TYPixel←
 FormatBayerRGGB16 = 0x0110002F, TYPixelFormatRGB8 = 0x02180014,
 TYPixelFormatBGR8 = 0x02180015, TYPixelFormatYUV422 8 = 0x02100032, TYPixelFormatYUV422 ↔
  8 UYVY = 0x0210001F, TYPixelFormatYCbCr420 8 YY CbCr Planar = 0x020C0113,
 TYPixelFormatYCbCr420 8 YY CrCb Planar = 0x020C0115, TYPixelFormatYCbCr420 8 YY CbCr←
 _Semiplanar = 0x020C0112, TYPixelFormatYCbCr420_8_YY_CrCb_Semiplanar = 0x020C0114, TY\leftarrow
 PixelFormatCoord3D C16 = 0x011000B8,
 TYPixelFormatCoord3D ABC16 = 0x023000B9, TYPixelFormatCoord3D ABC32f = 0x026000C0, TY←
 PixelFormatJPEG = 0x82180015, TYPixelFormatToflRFourGroupMono16 = 0x81400016,
 TYPixelFormatInvalid = 0xFFFFFFF }
    pixel format definitions

    enum TY PIXEL BITS LIST: uint32 t {

 TY_PIXEL_8BIT = 0x1 << 28, TY_PIXEL_16BIT = 0x2 << 28, TY_PIXEL_24BIT = 0x3 << 28, TY_PIX\leftarrow
 EL 32BIT = 0x4 << 28,
```

```
TY_PIXEL_10BIT = 0x5 << 28, TY_PIXEL_12BIT = 0x6 << 28, TY_PIXEL_14BIT = 0x7 << 28, TY_PI
   XEL 48BIT = (uint32 t)0x8 << 28,
   TY_PIXEL_64BIT = (uint32_t)0xa << 28 }
enum TY PIXEL FORMAT LIST: uint32 t {
   TY_PIXEL_FORMAT_UNDEFINED = 0, TY_PIXEL_FORMAT_MONO = (TY_PIXEL_8BIT | (0x0 << 24)),
   TY PIXEL FORMAT BAYER8GB = (TY PIXEL 8BIT | (0x1 << 24)), TY PIXEL FORMAT BAYER8BG =
   (TY PIXEL 8BIT | (0x2 << 24)),
   TY PIXEL FORMAT BAYER8GR = (TY PIXEL 8BIT | (0x3 << 24)), TY PIXEL FORMAT BAYER8RG
   = (TY PIXEL 8BIT | (0x4 << 24)), TY PIXEL FORMAT BAYER8GRBG = TY PIXEL FORMAT BAY↔
   ER8GB, TY_PIXEL_FORMAT_BAYER8RGGB = TY_PIXEL_FORMAT_BAYER8BG,
   TY PIXEL FORMAT BAYER8GBRG = TY PIXEL FORMAT BAYER8GR, TY PIXEL FORMAT BAY
   ER8BGGR = TY PIXEL FORMAT BAYER8RG, TY PIXEL FORMAT CSI MONO10 = (TY PIXEL 10BIT
   |(0x0 << 24)), TY PIXEL FORMAT CSI BAYER10GRBG = (TY PIXEL 10BIT |(0x1 << 24)),
   TY_PIXEL_FORMAT_CSI_BAYER10RGGB = (TY_PIXEL_10BIT | (0x2 << 24)), TY_PIXEL_FORMAT_CSI_BAYER10GBRG
   = (TY\_PIXEL\_10BIT | (0x3 << 24)), TY\_PIXEL\_FORMAT\_CSI\_BAYER10BGGR = (TY\_PIXEL\_10BIT | (0x4))
   << 24)), TY PIXEL FORMAT CSI MONO12 = (TY PIXEL 12BIT | (0x0 << 24)),
   TY PIXEL FORMAT CSI BAYER12GRBG = (TY PIXEL 12BIT | (0x1 << 24)), TY PIXEL FORMAT CSI BAYER12RGGB
   = (TY_PIXEL_12BIT | (0x2 << 24)), TY_PIXEL_FORMAT_CSI_BAYER12GBRG = (TY_PIXEL_12BIT | (0x3
   << 24)), TY PIXEL FORMAT CSI BAYER12BGGR = (TY PIXEL 12BIT | (0x4 << 24)),
   TY PIXEL FORMAT DEPTH16 = (TY PIXEL 16BIT | (0x0 << 24)), TY PIXEL FORMAT YVYU =
   (TY PIXEL 16BIT | (0x1 << 24)), TY PIXEL FORMAT YUYV = (TY PIXEL 16BIT | (0x2 << 24)),
   TY_{PIXEL}_{FORMAT\_MONO16} = (TY_{PIXEL}_{16BIT} | (0x3 << 24)),
   TY_PIXEL_FORMAT_TOF_IR_MONO16 = (TY_PIXEL_64BIT | (0x4 << 24)), TY_PIXEL_FORMAT_RGB
   = (TY_PIXEL_24BIT \mid (0x0 << 24)), TY_PIXEL_FORMAT_BGR = (TY_PIXEL_24BIT \mid (0x1 << 24)),
   TY_PIXEL_FORMAT_JPEG = (TY_PIXEL_24BIT | (0x2 << 24)),
   TY_PIXEL_FORMAT_MJPG = (TY_PIXEL_24BIT \mid (0x3 << 24)), TY_PIXEL_FORMAT_RGB48 = (T\leftrightarrow
   Y PIXEL 48BIT \mid (0x0 << 24)), TY PIXEL FORMAT BGR48 = (TY PIXEL 48BIT \mid (0x1 << 24)),
   TY PIXEL FORMAT XYZ48 = (TY PIXEL 48BIT | (0x2 << 24)) }
         pixel format definitions

    enum TY RESOLUTION MODE LIST: uint32 t {

   TY_RESOLUTION_MODE_160x100 = (160 << 12) + 100, TY_RESOLUTION_MODE_160x120 = (160 << 12) + 120,
   TY_RESOLUTION_MODE_240x320 = (240 << 12) + 320, TY_RESOLUTION_MODE_320x180 = (320 << 12) + 180, TY_RESOLUTION_MODE_320 = (320 << 12) + 180, TY_RESOLUTION_MODE_320 = (320 <<
   TY_RESOLUTION_MODE_320x200 = (320 << 12) + 200, TY_RESOLUTION_MODE_320x240 = (320 << 12) + 240,
   TY RESOLUTION MODE 480 \times 640 = (480 <<12) + 640, TY RESOLUTION MODE 640 \times 360 = (640 <<12) + 360,
   TY RESOLUTION MODE 640 \times 400 = (640 <<12) + 400, TY RESOLUTION MODE 640 \times 480 = (640 <<12) + 480,
   TY RESOLUTION MODE 960 \times 1280 = (960 < <12) + 1280, TY RESOLUTION MODE 1280 \times 720
   (1280 << 12) + 720
   TY RESOLUTION MODE 1280 \times 800 = (1280 < < 12) + 800. TY RESOLUTION MODE 1280 \times 960 = 1280 \times 
   (1280<<12)+960, TY_RESOLUTION_MODE_1600x1200 = (1600<<12)+1200, TY_RESOLUTION_MODE_800x600
   = (800 << 12) +600,
   TY_RESOLUTION_MODE_1920x1080 = (1920<<12)+1080, TY_RESOLUTION_MODE_2560x1920 =
   (2560<<12)+1920, TY RESOLUTION MODE 2592x1944 = (2592<<12)+1944, TY RESOLUTION MODE 1920x1440
   = (1920 << 12) + 1440,
   TY_RESOLUTION_MODE_240x96 = (240<<12)+96, TY_RESOLUTION_MODE_2048x1536 = (2048<<12)+1536
   }
         predefined resolution list
enum TY_IMAGE_MODE_LIST : uint32_t {
   TY DECLARE IMAGE MODE1 = (MONO), TY DECLARE IMAGE MODE1 = (MONO), TY DECLARE -
   IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO),
   TY_DECLARE_IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO), TY_DECLARE_
   IMAGE MODE1 = (MONO), TY DECLARE IMAGE MODE1 = (MONO),
   TY DECLARE IMAGE MODE1 = (MONO). TY DECLARE IMAGE MODE1 = (MONO). TY DECLARE \leftrightarrow
   IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO),
   TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE -
   IMAGE MODE1 = (MONO), TY_DECLARE_IMAGE_MODE1 = (MONO),
   TY DECLARE IMAGE MODE1 = (MONO), TY DECLARE IMAGE MODE1 = (MONO), TY DECLARE \
   IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO),
   TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE ↔
```

```
IMAGE MODE1 = (MONO), TY DECLARE IMAGE MODE1 = (MONO),
 TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE IMAGE MODE1 =(MONO), TY DECLARE -
 IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO),
 TY_DECLARE_IMAGE_MODE1 =(MONO), TY_DECLARE_IMAGE_MODE1 =(MONO) }
    Predefined Image Mode List image mode controls image resolution & format predefined image modes named like
    TY_IMAGE_MODE_MONO_160x120,TY_IMAGE_MODE_RGB_1280x960.
• enum TY TRIGGER_MODE_LIST : uint32_t {
 TY TRIGGER MODE OFF = 0, TY TRIGGER MODE SLAVE = 1, TY TRIGGER MODE M SIG = 2,
 TY TRIGGER MODE M PER = 3,
 TY TRIGGER MODE SIG PASS = 18, TY TRIGGER MODE PER PASS = 19, TY TRIGGER MODE ↔
 TIMER LIST = 20, TY_TRIGGER MODE TIMER PERIOD = 21,
 TY TRIGGER MODE28 = 28, TY TRIGGER MODE29 = 29, TY TRIGGER MODE PER PASS2 = 30,
 TY_TRIGGER_WORK_MODE31 = 31,
 TY_TRIGGER_MODE_SIG_LASER = 34 }

    enum TY TIME SYNC TYPE LIST: uint32 t {

 TY TIME SYNC TYPE NONE = 0, TY TIME SYNC TYPE HOST = 1, TY TIME SYNC TYPE NTP = 2,
 TY_TIME_SYNC_TYPE_PTP = 3,
 TY_TIME_SYNC_TYPE_CAN = 4, TY_TIME_SYNC_TYPE_PTP_MASTER = 5 }
    type of time sync
enum TY LOG LEVEL LIST {
 TY_LOG_LEVEL_VERBOSE = 1, TY_LOG_LEVEL_DEBUG = 2, TY_LOG_LEVEL_INFO = 3, TY_LOG ←
 LEVEL WARNING = 4,
 TY LOG LEVEL ERROR = 5, TY LOG LEVEL NEVER = 9 }

    enum TY VISIBILITY TYPE { BEGINNER = 0, EXPERT = 1, GURU = 2, INVLISIBLE = 3 }

• enum { TY PATTERN SINE TYPE = 0, TY PATTERN GRAY TYPE, TY PATTERN BIN TYPE, TY ←
 PATTERN EMPTY TYPE = 0xffffffff }
enum { TY_NORMAL_PHASE_TYPE = 0, TY_REFER_PHASE_TYPE }
enum TY IMU FPS LIST { TY IMU FPS 100HZ = 0, TY IMU FPS 200HZ, TY IMU FPS 400HZ }
```

Variables

- typedef enum
- typedef **TY_DO_5V** = 1
- typedef **TY_DO_12V** = 2
- typedef TY_E_VOLT_T_LIST
- typedef TY DO HIGH = 1
- typedef TY_DO_PWM = 2
- typedef TY_DO_CAM_TRIG = 3
- typedef TY_E_DO_MODE_LIST
- typedef TY DI NE INT = 1
- typedef **TY_DI_PE_INT** = 2
- typedef TY_E_DI_MODE_LIST
- typedef TY_DI_INT_TRIG_CAP = 1
- typedef TY DI INT EVENT = 2
- typedef TY_E_DI_INT_ACTION_LIST
- typedef TY_TEMPERATURE_RIGHT = 1
- typedef **TY_TEMPERATURE_COLOR** = 2
- typedef TY_TEMPERATURE_CPU = 3
- typedef TY_TEMPERATURE_MAIN_BOARD = 4
- typedef TY_TEMPERATURE_ID_LIST

5.3.1 Detailed Description

TYDefs.h includes camera control and data receiving data definitions which supports configuration for image resolution, frame rate, exposure time, gain, working mode,etc.

5.3 TYDefs.h File Reference 121

5.3.2 Macro Definition Documentation

5.3.2.1 TY_DECLARE_IMAGE_MODE1

```
#define TY_DECLARE_IMAGE_MODE1(
                   pix )
Value:
              TY_DECLARE_IMAGE_MODE0(pix, 160x100),
               TY_DECLARE_IMAGE_MODE0(pix, 160x120),
               TY_DECLARE_IMAGE_MODE0(pix, 320x180),
               TY_DECLARE_IMAGE_MODE0(pix, 320x200),
               TY_DECLARE_IMAGE_MODEO(pix, 320x240),
TY_DECLARE_IMAGE_MODEO(pix, 320x240),
TY_DECLARE_IMAGE_MODEO(pix, 480x640),
               TY_DECLARE_IMAGE_MODE0(pix, 640x360),
               TY_DECLARE_IMAGE_MODE0(pix, 640x400),
               TY_DECLARE_IMAGE_MODE0(pix, 640x480),
               TY_DECLARE_IMAGE_MODE0(pix, 960x1280),
TY_DECLARE_IMAGE_MODE0(pix, 1280x720),
               TY_DECLARE_IMAGE_MODEO(pix, 1280x960),
TY_DECLARE_IMAGE_MODEO(pix, 1280x800),
               TY_DECLARE_IMAGE_MODE0(pix, 1600x1200),
               TY_DECLARE_IMAGE_MODE0(pix, 800x600), \
               TY_DECLARE_IMAGE_MODE0(pix, 1920x1080),
```

TY_DECLARE_IMAGE_MODEO(pix, 2560x1920),
TY_DECLARE_IMAGE_MODEO(pix, 2592x1944),
TY_DECLARE_IMAGE_MODEO(pix, 1920x1440),
TY_DECLARE_IMAGE_MODEO(pix, 2048x1536),
TY_DECLARE_IMAGE_MODEO(pix, 240x96)

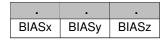
Definition at line 637 of file TYDefs.h.

5.3.3 Typedef Documentation

5.3.3.1 TY_ACC_BIAS

```
typedef struct TY_ACC_BIAS TY_ACC_BIAS
```

a 3x3 matrix



5.3.3.2 TY_ACC_MISALIGNMENT

```
typedef struct TY_ACC_MISALIGNMENT TY_ACC_MISALIGNMENT
a 3x3 matrix
|.|.|.|
```

	•	•
1	-GAMAyz	GAMAzy
GAMAxz	1	-GAMAzx
-GAMAxy	GAMAyx	1

5.3.3.3 TY_ACC_SCALE

 ${\tt typedef \ struct \ TY_ACC_SCALE \ TY_ACC_SCALE}$

a 3x3 matrix

•		•
SCALEx	0	0
0	SCALEy	0
0	0	SCALEz

5.3.3.4 TY_ACCESS_MODE_LIST

typedef enum TY_ACCESS_MODE_LIST TY_ACCESS_MODE_LIST

Indicate a feature is readable or writable

See also

TYGetFeatureInfo

5.3.3.5 TY_BYTEARRAY_ATTR

typedef struct TY_BYTEARRAY_ATTR TY_BYTEARRAY_ATTR

byte array data structure

See also

TYGetByteArray

5.3.3.6 TY_CAMERA_CALIB_INFO

typedef struct TY_CAMERA_CALIB_INFO TY_CAMERA_CALIB_INFO

camera 's cailbration data

See also

TYGetStruct

5.3.3.7 TY_CAMERA_DISTORTION

```
typedef struct TY_CAMERA_DISTORTION TY_CAMERA_DISTORTION
```

camera distortion parameters

See also

```
TYGetStruct Usage:
```

```
TY_CAMERA_DISTORTION distortion;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_DISTORTION, &distortion, sizeof(distortion));
```

5.3.3.8 TY_CAMERA_EXTRINSIC

```
typedef struct TY_CAMERA_EXTRINSIC TY_CAMERA_EXTRINSIC
```

a 4x4 matrix

•	•	•	-
r11	r12	r13	t1
r21	r22	r23	t2
r31	r32	r33	t3
0	0	0	1

See also

```
TYGetStruct Usage:
```

```
TY_CAMERA_EXTRINSIC extrinsic;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_EXTRINSIC, &extrinsic, sizeof(extrinsic));
```

5.3.3.9 TY_CAMERA_INTRINSIC

```
typedef struct TY_CAMERA_INTRINSIC TY_CAMERA_INTRINSIC
```

a 3x3 matrix

	-	
fx	0	сх
0	fy	су
0	0	1

See also

TYGetStruct Usage:

TY_CAMERA_INTRINSIC intrinsic;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_INTRINSIC, &intrinsic, sizeof(intrinsic));

5.3.3.10 TY_CAMERA_ROTATION

typedef struct TY_CAMERA_ROTATION TY_CAMERA_ROTATION

a 3x3 matrix

		•
r00	r01	r02
r10	r11	r12
r20	r21	r22

See also

TYGetStruct Usage:

TY_CAMERA_ROTATION rotation;
TYGetStruct(hDevice, some_compoent, TY_STRUCT_CAM_ROTATION, &rotation, sizeof(rotation));

5.3.3.11 TY_CAMERA_TO_IMU

typedef struct TY_CAMERA_TO_IMU TY_CAMERA_TO_IMU

a 4x4 matrix

r11	r12	r13	t1
r21	r22	r23	t2
r31	r32	r33	t3
0	0	0	1

5.3.3.12 TY_COMPONENT_ID

typedef uint32_t TY_COMPONENT_ID

component unique id

See also

TY_DEVICE_COMPONENT_LIST

Definition at line 191 of file TYDefs.h.

5.3.3.13 TY_DEVICE_BASE_INFO

typedef struct TY_DEVICE_BASE_INFO TY_DEVICE_BASE_INFO

See also

TYGetDeviceList

5.3.3.14 TY_DEVICE_COMPONENT_LIST

typedef enum TY_DEVICE_COMPONENT_LIST TY_DEVICE_COMPONENT_LIST

@breif Device Component list A device contains several component. Each component can be controlled by its own features, such as image width, exposure time, etc..

See also

To Know how to get feature information please refer to sample code DumpAllFeatures

5.3.3.15 TY_ENUM_ENTRY

typedef struct TY_ENUM_ENTRY TY_ENUM_ENTRY

enum feature entry information

See also

TYGetEnumEntryInfo

5.3.3.16 TY_FEATURE_ID

typedef uint32_t TY_FEATURE_ID

feature unique id

See also

TY_FEATURE_ID_LIST

Definition at line 399 of file TYDefs.h.

5.3.3.17 TY_FLOAT_RANGE

typedef struct TY_FLOAT_RANGE TY_FLOAT_RANGE

float range data structure

See also

TYGetFloatRange

5.3.3.18 TY_GYRO_BIAS

typedef struct TY_GYRO_BIAS TY_GYRO_BIAS

a 3x3 matrix

BIASx	BIASy	BIASz

5.3.3.19 TY_GYRO_MISALIGNMENT

typedef struct TY_GYRO_MISALIGNMENT TY_GYRO_MISALIGNMENT

a 3x3 matrix

		•
1	-ALPHAyz	ALPHAzy
0	1	-ALPHAzx
0	0	1

5.3.3.20 TY_GYRO_SCALE

typedef struct TY_GYRO_SCALE TY_GYRO_SCALE

a 3x3 matrix

		•
SCALEx	0	0
0	SCALEy	0
0	0	SCALEz

5.3.3.21 TY_INTERFACE_INFO

typedef struct TY_INTERFACE_INFO TY_INTERFACE_INFO

See also

TYGetInterfaceList

5.3.3.22 TY_INTERFACE_TYPE_LIST

typedef enum TY_INTERFACE_TYPE_LIST TY_INTERFACE_TYPE_LIST

Interface type definition

See also

TYGetInterfaceList

5.3.3.23 TY_PIXEL_BITS_LIST

typedef enum TY_PIXEL_BITS_LIST TY_PIXEL_BITS_LIST

Pixel size type definitions to define the pixel size in bits

See also

TY_PIXEL_FORMAT_LIST

5.3.3.24 TY_TRIGGER_MODE_LIST

```
typedef enum TY_TRIGGER_MODE_LIST TY_TRIGGER_MODE_LIST
```

See also

refer to sample SimpleView TriggerMode for detail usage

5.3.4 Enumeration Type Documentation

5.3.4.1 TY_ACCESS_MODE_LIST

```
enum TY_ACCESS_MODE_LIST : uint32_t
```

Indicate a feature is readable or writable

See also

TYGetFeatureInfo

Definition at line 453 of file TYDefs.h.

5.3.4.2 TY_DEVICE_COMPONENT_LIST

```
enum TY_DEVICE_COMPONENT_LIST : uint32_t
```

@breif Device Component list A device contains several component. Each component can be controlled by its own features, such as image width, exposure time, etc..

See also

To Know how to get feature information please refer to sample code DumpAllFeatures

TY_COMPONENT_DEVICE	Abstract component stands for whole device, always enabled.
TY_COMPONENT_DEPTH_CAM	Depth camera.
TY_COMPONENT_IR_CAM_LEFT	Left IR camera.
TY_COMPONENT_IR_CAM_RIGHT	Right IR camera.
TY_COMPONENT_RGB_CAM_LEFT	Left RGB camera.
TY_COMPONENT_RGB_CAM_RIGHT	Right RGB camera.
TY_COMPONENT_LASER	Laser.
TY_COMPONENT_IMU	Inertial Measurement Unit.
TY_COMPONENT_BRIGHT_HISTO	virtual component for brightness histogram of ir
TY_COMPONENT_STORAGE	virtual component for device storage
TY_COMPONENT_RGB_CAM	Some device has only one RGB camera, map it to letenerated by Doxyger

5.3 TYDefs.h File Reference 129

Definition at line 176 of file TYDefs.h.

5.3.4.3 TY_FEATURE_ID_LIST

```
enum TY_FEATURE_ID_LIST : uint32_t
```

feature for component definitions

TY_STRUCT_CAM_INTRINSIC	see TY_CAMERA_INTRINSIC
TY_STRUCT_EXTRINSIC_TO_DEPTH	extrinsic between depth cam and current component, see TY_CAMERA_EXTRINSIC
TY_STRUCT_EXTRINSIC_TO_IR_LEFT	extrinsic between left IR and current compoent, see TY_CAMERA_EXTRINSIC
TY_STRUCT_CAM_RECTIFIED_ROTATION	see TY_CAMERA_ROTATION
TY_STRUCT_CAM_DISTORTION	see TY_CAMERA_DISTORTION
TY_STRUCT_CAM_CALIB_DATA	see TY_CAMERA_CALIB_INFO
TY_STRUCT_CAM_RECTIFIED_INTRI	the rectified intrinsic. see TY_CAMERA_INTRINSIC
TY_BYTEARRAY_CUSTOM_BLOCK	used for reading/writing custom block
TY_BYTEARRAY_ISP_BLOCK	used for reading/writing fpn block
TY_INT_PACKET_DELAY	microseconds
TY_INT_NTP_SERVER_IP	Ntp server IP.
TY_INT_LINK_CMD_TIMEOUT	milliseconds
TY_STRUCT_CAM_STATISTICS	statistical information, see TY_CAMERA_STATISTICS
TY_INT_WIDTH	Image width.
TY_INT_HEIGHT	Image height.
TY_ENUM_IMAGE_MODE	Resolution-PixelFromat mode, see TY_IMAGE_MODE_LIST.
TY_FLOAT_SCALE_UNIT	scale unit depth image is uint16 pixel format with default millimeter unit ,for some device can output Sub-millimeter accuracy data the acutal depth (mm)= PixelValue * ScaleUnit
TY_ENUM_TRIGGER_POL	Trigger POL, see TY_TRIGGER_POL_LIST.
TY_INT_FRAME_PER_TRIGGER	Number of frames captured per trigger.
TY_ENUM_USER_SET_CURRENT	Set the configuration set to be used as the default startup set. Write & Read
TY_ENUM_USER_SET_SELECTOR	UserSetSelector. Indicates the currently active user configuration profile. Read Only
TY_CMD_USER_SET_LOAD	Selects the configuration set to load, save, or configure. Write & Read
TY_CMD_USER_SET_SAVE	Stores the selected configuration in the camera's volatile memory and sets it as the temporary active configuration. Write Only
TY_STRING_USER_SET_DESCRIPTION	Saves the active configuration set to the chosen user set(set by TY_ENUM_USER_SET_SELECTOR). Write Only
TY_STRUCT_TRIGGER_PARAM	param of trigger, see TY_TRIGGER_PARAM Description of the selected User Set content. Write & Read
TY_STRUCT_TRIGGER_PARAM_EX	param of trigger, see TY_TRIGGER_PARAM_EX
TY_STRUCT_TRIGGER_TIMER_LIST	param of trigger mode 20, see TY_TRIGGER_TIMER_LIST

TY_STRUCT_TRIGGER_TIMER_PERIOD	param of trigger mode 21, see TY_TRIGGER_TIMER_PERIOD
TY_BOOL_KEEP_ALIVE_ONOFF	Keep Alive switch.
TY_INT_KEEP_ALIVE_TIMEOUT	Keep Alive timeout.
TY_BOOL_CMOS_SYNC	Cmos sync switch.
TY_INT_TRIGGER_DELAY_US	Trigger delay time, in microseconds.
TY_BOOL_TRIGGER_OUT_IO	Trigger out IO.
TY_INT_TRIGGER_DURATION_US	Trigger duration time, in microseconds.
TY_ENUM_STREAM_ASYNC	stream async switch, see TY_STREAM_ASYNC_MODE
TY_INT_CAPTURE_TIME_US	capture time in multi-ir
TY_ENUM_TIME_SYNC_TYPE	see TY_TIME_SYNC_TYPE
TY_BOOL_TIME_SYNC_READY	time sync done status
TY_BOOL_IR_FLASHLIGHT	Enable switch for floodlight used in ir component.
TY_INT_IR_FLASHLIGHT_INTENSITY	ir component flashlight intensity level
TY_STRUCT_DO0_WORKMODE	DO_0 workmode, see TY_DO_WORKMODE.
TY_STRUCT_DI0_WORKMODE	DI_0 workmode, see TY_DI_WORKMODE.
TY_STRUCT_DO1_WORKMODE	DO_1 workmode, see TY_DO_WORKMODE.
TY_STRUCT_DI1_WORKMODE	DI_1 workmode, see TY_DI_WORKMODE.
TY_STRUCT_DO2_WORKMODE	DO_2 workmode, see TY_DO_WORKMODE.
TY_STRUCT_DI2_WORKMODE	DI_2 workmode, see TY_DI_WORKMODE.
TY_BOOL_RGB_FLASHLIGHT	Enable switch for floodlight used in rgb component.
TY_INT_RGB_FLASHLIGHT_INTENSITY	rgb component flashlight intensity level
TY_BOOL_AUTO_EXPOSURE	Auto exposure switch.
TY_INT_EXPOSURE_TIME	Exposure time.
TY_BOOL_AUTO_GAIN	Auto gain switch.
TY_INT_GAIN	Sensor Gain.
TY_BOOL_AUTO_AWB	Auto white balance.
TY_STRUCT_AEC_ROI	region of aec statistics, see TY_AEC_ROI_PARAM
TY_INT_TOF_HDR_RATIO	tof sensor hdr ratio for depth
TY_INT_TOF_JITTER_THRESHOLD	tof jitter threshold for depth
TY_FLOAT_EXPOSURE_TIME_US	the exposure time, unit: us
TY_INT_LASER_POWER	Laser power level.
TY_BOOL_LASER_AUTO_CTRL	Laser auto ctrl.
TY_STRUCT_LASER_ENABLE_BY_IDX	Laser enable by device index.
TY_STRUCT_LASER_POWER_BY_IDX	Laser power by device index.
TY_STRUCT_FLOOD_ENABLE_BY_IDX	Flood enable by device index.
TY_STRUCT_FLOOD_POWER_BY_IDX	Flood power by device index.
TY_BOOL_UNDISTORTION	Output undistorted image.
TY_BOOL_BRIGHTNESS_HISTOGRAM	Output bright histogram.
TY_BOOL_DEPTH_POSTPROC	Do depth image postproc.
TY_INT_R_GAIN	Gain of R channel.
TY_INT_G_GAIN	Gain of G channel.
TY_INT_B_GAIN	Gain of B channel.
TY_INT_ANALOG_GAIN	Analog gain.
TY_BOOL_HDR	HDR func enable/disable.
TY_BYTEARRAY_HDR_PARAMETER	HDR parameters.
TY_BOOL_IMU_DATA_ONOFF	AE target y. IMU Data Onoff

TY_STRUCT_IMU_ACC_BIAS	IMU acc bias matrix, see TY_ACC_BIAS.
TY_STRUCT_IMU_ACC_MISALIGNMENT	IMU acc misalignment matrix, see
	TY_ACC_MISALIGNMENT.
TY_STRUCT_IMU_ACC_SCALE	IMU acc scale matrix, see TY_ACC_SCALE.
TY_STRUCT_IMU_GYRO_BIAS	IMU gyro bias matrix, see TY_GYRO_BIAS.
TY_STRUCT_IMU_GYRO_MISALIGNMENT	IMU gyro misalignment matrix, see
TY STRUCT IMU GYRO SCALE	TY_GYRO_MISALIGNMENT. IMU gyro scale matrix, see TY_GYRO_SCALE.
TY_STRUCT_IMU_CAM_TO_IMU	IMU camera to imu matrix, see TY_CAMERA_TO_IMU.
TY_STRUCT_IMIO_CAMI_TO_IMIO TY_ENUM_IMU_FPS	IMU fps, see TY_IMU_FPS_LIST.
TY_INT_SGBM_IMAGE_NUM	SGBM image channel num.
TY_INT_SGBM_DISPARITY_NUM	SGBM disparity num.
TY_INT_SGBM_DISPARITY_OFFSET	SGBM disparity offset.
TY_INT_SGBM_MATCH_WIN_HEIGHT	SGBM match window height.
	-
TY_INT_SGBM_SEMI_PARAM_P1	SGBM semi global param p2
TY_INT_SGBM_SEMI_PARAM_P2	SGBM semi global param p2.
TY_INT_SGBM_UNIQUE_FACTOR	SGBM uniqueness factor param.
TY_INT_SGBM_UNIQUE_ABSDIFF	SGBM uniqueness min absolute diff.
TY_INT_SGBM_UNIQUE_MAX_COST	SGBM uniqueness max cost param.
TY_BOOL_SGBM_HFILTER_HALF_WIN	SGBM enable half window size.
TY_INT_SGBM_MATCH_WIN_WIDTH	SGBM match window width. SGBM enable median filter.
TY_BOOL_SGBM_MEDFILTER TY_BOOL_SGBM_LRC	SGBM enable left right consist check.
TY INT SGBM LRC DIFF	SGBM max diff.
TY INT SGBM MEDFILTER THRESH	SGBM median filter thresh.
TY_INT_SGBM_SEMI_PARAM_P1_SCALE	SGBM semi global param p1 scale.
TY_INT_SGPM_PHASE_NUM	Phase num to calc a depth.
TY_INT_SGPM_NORMAL_PHASE_SCALE	phase scale when calc a depth
TY_INT_SGPM_NORMAL_PHASE_OFFSET	Phase offset when calc a depth.
TY_INT_SGPM_REF_PHASE_SCALE	Reference Phase scale when calc a depth.
TY_INT_SGPM_REF_PHASE_OFFSET	Reference Phase offset when calc a depth.
TY_FLOAT_SGPM_EPI_HS	Epipolar Constraint pattern scale.
TY_INT_SGPM_EPI_HF	Epipolar Constraint pattern offset.
TY_BOOL_SGPM_EPI_EN	Epipolar Constraint enable.
TY INT SGPM EPI CH0	Epipolar Constraint enable. Epipolar Constraint channel0.
TY_INT_SGPM_EPI_CH1	Epipolar Constraint channel1.
TY INT SGPM EPI THRESH	Epipolar Constraint charmer: Epipolar Constraint thresh.
TY BOOL SGPM ORDER FILTER EN	Phase order filter enable.
TY INT SGPM ORDER FILTER CHN	Phase order filter channel.
TY_INT_DEPTH_MIN_MM	min depth in mm output
TY_INT_DEPTH_MAX_MM	max depth in mm output
TY INT SGBM TEXTURE OFFSET	texture filter value offset
TY_INT_SGBM_TEXTURE_THRESH	texture filter threshold
TY_STRUCT_PHC_GROUP_ATTR	Phase compute group attribute.
TY_ENUM_DEPTH_QUALITY	the quality of generated depth, see TY_DEPTH_QUALITY
TY_INT_FILTER_THRESHOLD	the threshold of the noise filter, 0 for disabled
TY_INT_TOF_CHANNEL	the frequency channel of tof
TY_INT_TOF_MODULATION_THRESHOLD	the threshold of the tof modulation

Enumerator

TY_STRUCT_TOF_FREQ	the frequency of tof, see TY_TOF_FREQ
TY_BOOL_TOF_ANTI_INTERFERENCE	cooperation if multi-device used
TY_INT_TOF_ANTI_SUNLIGHT_INDEX	the index of anti-sunlight
TY_INT_MAX_SPECKLE_SIZE	the max size of speckle
TY_INT_MAX_SPECKLE_DIFF	the max diff of speckle

Definition at line 211 of file TYDefs.h.

5.3.4.4 TY_INTERFACE_TYPE_LIST

```
enum TY_INTERFACE_TYPE_LIST : uint32_t
```

Interface type definition

See also

TYGetInterfaceList

Definition at line 440 of file TYDefs.h.

5.3.4.5 TY_PIXEL_BITS_LIST

```
enum TY_PIXEL_BITS_LIST : uint32_t
```

Pixel size type definitions to define the pixel size in bits

See also

TY_PIXEL_FORMAT_LIST

Definition at line 545 of file TYDefs.h.

5.3.4.6 TY_PIXEL_FORMAT_LIST

```
enum TY_PIXEL_FORMAT_LIST : uint32_t
```

pixel format definitions

Enumerator

TY_PIXEL_FORMAT_MONO	0x10000000
TY_PIXEL_FORMAT_BAYER8GB	0x11000000
TY_PIXEL_FORMAT_BAYER8BG	0x12000000
TY_PIXEL_FORMAT_BAYER8GR	0x13000000
TY_PIXEL_FORMAT_BAYER8RG	0x14000000
TY_PIXEL_FORMAT_CSI_MONO10	0x50000000
TY_PIXEL_FORMAT_CSI_BAYER10GRBG	0x51000000
TY_PIXEL_FORMAT_CSI_BAYER10RGGB	0x52000000
TY_PIXEL_FORMAT_CSI_BAYER10GBRG	0x53000000
TY_PIXEL_FORMAT_CSI_BAYER10BGGR	0x54000000
TY_PIXEL_FORMAT_CSI_MONO12	0x60000000
TY_PIXEL_FORMAT_CSI_BAYER12GRBG	0x61000000
TY_PIXEL_FORMAT_CSI_BAYER12RGGB	0x62000000
TY_PIXEL_FORMAT_CSI_BAYER12GBRG	0x63000000
TY_PIXEL_FORMAT_CSI_BAYER12BGGR	0x64000000
TY_PIXEL_FORMAT_DEPTH16	0x20000000
TY_PIXEL_FORMAT_YVYU	0x21000000, yvyu422
TY_PIXEL_FORMAT_YUYV	0x22000000, yuyv422
TY_PIXEL_FORMAT_MONO16	0x23000000,
TY_PIXEL_FORMAT_TOF_IR_MONO16	0xa4000000,
TY_PIXEL_FORMAT_RGB	0x30000000
TY_PIXEL_FORMAT_BGR	0x31000000
TY_PIXEL_FORMAT_JPEG	0x32000000
TY_PIXEL_FORMAT_MJPG	0x33000000
TY_PIXEL_FORMAT_RGB48	0x80000000
TY_PIXEL_FORMAT_BGR48	0x81000000
TY_PIXEL_FORMAT_XYZ48	0x82000000

Definition at line 563 of file TYDefs.h.

5.3.4.7 TY_RESOLUTION_MODE_LIST

enum TY_RESOLUTION_MODE_LIST : uint32_t

predefined resolution list

TY_RESOLUTION_MODE_160x100	0x000a0078
TY_RESOLUTION_MODE_160x120	0x000a0078
TY_RESOLUTION_MODE_240x320	0x000f0140
TY_RESOLUTION_MODE_320x180	0x001400b4
TY_RESOLUTION_MODE_320x200	0x001400c8
TY_RESOLUTION_MODE_320x240	0x001400f0
TY_RESOLUTION_MODE_480x640	0x001e0280
TY_RESOLUTION_MODE_640x360	0x00280168

Enumerator

TY_RESOLUTION_MODE_640x400	0x00280190
TY_RESOLUTION_MODE_640x480	0x002801e0
TY_RESOLUTION_MODE_960x1280	0x003c0500
TY_RESOLUTION_MODE_1280x720	0x005002d0
TY_RESOLUTION_MODE_1280x800	0x00500320
TY_RESOLUTION_MODE_1280x960	0x005003c0
TY_RESOLUTION_MODE_1600x1200	0x006404b0
TY_RESOLUTION_MODE_800x600	0x00320258
TY_RESOLUTION_MODE_1920x1080	0x00780438
TY_RESOLUTION_MODE_2560x1920	0x00a00780
TY_RESOLUTION_MODE_2592x1944	0x00a20798
TY_RESOLUTION_MODE_1920x1440	0x007805a0
TY_RESOLUTION_MODE_240x96	0x000f0060
TY_RESOLUTION_MODE_2048x1536	0x00800600

Definition at line 607 of file TYDefs.h.

5.3.4.8 TY_TRIGGER_MODE_LIST

```
enum TY_TRIGGER_MODE_LIST : uint32_t
```

See also

refer to sample SimpleView_TriggerMode for detail usage

Enumerator

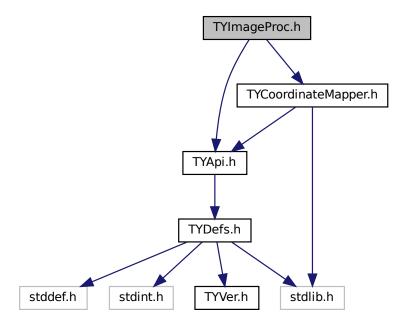
TY_TRIGGER_MODE_OFF	not trigger mode, continuous mode
TY_TRIGGER_MODE_SLAVE	slave mode, receive soft/hardware triggers
TY_TRIGGER_MODE_M_SIG	master mode 1, sending one trigger signal once received a soft trigger
TY_TRIGGER_MODE_M_PER	master mode 2, periodic sending one trigger signals, 'fps' param
	should be set
TY_TRIGGER_MODE_SIG_PASS	discard, using TY_TRIGGER_MODE28
TY_TRIGGER_MODE_PER_PASS	discard, using TY_TRIGGER_MODE29
TY_TRIGGER_MODE_PER_PASS2	trigger mode 30,Alternate output depth image/ir image

Definition at line 708 of file TYDefs.h.

5.4 TYImageProc.h File Reference

```
#include "TYApi.h"
#include "TYCoordinateMapper.h"
```

Include dependency graph for TYImageProc.h:



Classes

- struct DepthSpeckleFilterParameters
- struct DepthEnhenceParameters

Macros

- #define DepthSpeckleFilterParameters_Initializer {150, 64}
- #define DepthEnhenceParameters_Initializer {10, 20, 10, 0.1f}

Functions

- TY CAPI TYImageProcesAcceEnable (bool en)
 - Image processing acceleration switch.
- TY_CAPI TYUndistortImage (const TY_CAMERA_CALIB_INFO *srcCalibInfo, const TY_IMAGE_DATA *srcImage, const TY_CAMERA_INTRINSIC *cameraNewIntrinsic, TY_IMAGE_DATA *dstImage)
 - Do image undistortion, only support $TY_PIXEL_FORMAT_MONO$, $TY_PIXEL_FORMAT_RGB$, $TY_PIXEL_FORMAT_RGB$.
- TY_CAPI TYDepthSpeckleFilter (TY_IMAGE_DATA *depthImage, const DepthSpeckleFilterParameters *param)
 - Remove speckles on depth image.
- TY_CAPI TYDepthEnhenceFilter (const TY_IMAGE_DATA *depthImages, int imageNum, TY_IMAGE_DATA *guide, TY_IMAGE_DATA *output, const DepthEnhenceParameters *param)
 - Remove speckles on depth image.

5.4.1 Detailed Description

@breif Image post-process API

Copyright

Copyright(C)2016-2018 Percipio All Rights Reserved

5.4.2 Function Documentation

5.4.2.1 TYDepthEnhenceFilter()

Remove speckles on depth image.

Parameters

in	depthImage	Pointer to depth image array.
in	imageNum	Depth image array size.
in,out	guide	Guide image.
out	output	Output depth image.
in	param	Algorithm parameters.

Return values

TY_STATUS_OK	Succeed.
TY_STATUS_NULL_POINTER	Any depthImage, param, output or output->buffer is NULL.
TY_STATUS_INVALID_PARAMETER	imageNum >= 11 or imageNum <= 0, or any image invalid
TY_STATUS_OUT_OF_MEMORY	Output image not suitable.

5.4.2.2 TYDepthSpeckleFilter()

Remove speckles on depth image.

Parameters

in,out	depthImage	Depth image to be processed.	
in	param	Algorithm parameters.	

Return values

TY_ST	ATUS_OK	Succeed.
TY_STATUS_NULL_	_POINTER	Any depth, param or depth->buffer is NULL.
TY_STATUS_INVALID_PA	RAMETER	param->max_speckle_size <= 0 or param->max_speckle_diff <= 0

5.4.2.3 TYImageProcesAcceEnable()

```
TY_CAPI TYImageProcesAcceEnable ( bool en )
```

Image processing acceleration switch.

Parameters

in	en	Enable image process acceleration switch	
----	----	--	--

5.4.2.4 TYUndistortImage()

Do image undistortion, only support TY_PIXEL_FORMAT_MONO , TY_PIXEL_FORMAT_RGB, TY_PIXEL_FOR \longleftrightarrow MAT_BGR.

Parameters

in	srcCalibInfo	Image calibration data.
in	srcImage	Source image.
in	cameraNewIntrinsic	Expected new image intrinsic, will use srcCalibInfo for new image intrinsic if set
		to NULL.
out	dstlmage	Output image.

TY_STATUS_OK Succeed.

TY_STATUS_NULL_POINTER	Any srcCalibInfo, srcImage, dstImage, srcImage->buffer, dstImage->buffer is NULL.
TY_STATUS_INVALID_PARAMETER	Invalid srcImage->width, srcImage->height, dstImage->width, dstImage->height or unsupported pixel format.

Index

DepthEnhenceParameters, 7	TYDefs.h, 131
DepthSpeckleFilterParameters, 7	TY_BOOL_SGPM_ORDER_FILTER_EN
	TYDefs.h, 131
gz_header_s, 8	TY_BOOL_TIME_SYNC_READY
gzFile_s, 8	TYDefs.h, 130
3, -	TY_BOOL_TOF_ANTI_INTERFERENCE
pattern_bin_param, 8	
pattern_gray_param, 9	TYDefs.h, 132
pattern_sine_param, 9	TY_BOOL_TRIGGER_OUT_IO
pattern_oo_pattatin, •	TYDefs.h, 130
TY_ACC_BIAS, 10	TY_BOOL_UNDISTORTION
TYDefs.h, 121	TYDefs.h, 130
TY_ACC_MISALIGNMENT, 10	TY_BYTEARRAY_ATTR, 12
TYDefs.h, 121	TYDefs.h, 122
TY_ACC_SCALE, 11	unit_size, 12
	valid_size, 13
TYDefs.h, 122	TY_BYTEARRAY_CUSTOM_BLOCK
TY_ACCESS_MODE_LIST	TYDefs.h, 129
TYDefs.h, 122, 128	TY_BYTEARRAY_HDR_PARAMETER
TY_AEC_ROI_PARAM, 12	TYDefs.h, 130
TY_BOOL_AUTO_AWB	TY_BYTEARRAY_ISP_BLOCK
TYDefs.h, 130	TYDefs.h, 129
TY_BOOL_AUTO_EXPOSURE	
TYDefs.h, 130	TY_CAMERA_CALIB_INFO, 13
TY_BOOL_AUTO_GAIN	TYDefs.h, 122
TYDefs.h, 130	TY_CAMERA_DISTORTION, 14
TY_BOOL_BRIGHTNESS_HISTOGRAM	TYDefs.h, 123
TYDefs.h, 130	TY_CAMERA_EXTRINSIC, 14
TY_BOOL_CMOS_SYNC	TYDefs.h, 123
TYDefs.h, 130	TY_CAMERA_INTRINSIC, 15
TY_BOOL_DEPTH_POSTPROC	TYDefs.h, 123
TYDefs.h, 130	TY_CAMERA_ROTATION, 16
TY BOOL HDR	TYDefs.h, 124
TYDefs.h, 130	TY_CAMERA_STATISTICS, 16
TY_BOOL_IMU_DATA_ONOFF	TY_CAMERA_TO_IMU, 17
TYDefs.h, 130	TYDefs.h, 124
	TY_CMD_USER_SET_LOAD
TY_BOOL_IR_FLASHLIGHT	TYDefs.h, 129
TYDefs.h, 130	TY CMD USER SET SAVE
TY_BOOL_KEEP_ALIVE_ONOFF	
TYDefs.h, 130	TYDefs.h, 129
TY_BOOL_LASER_AUTO_CTRL	TY_COMPONENT_BRIGHT_HISTO
TYDefs.h, 130	TYDefs.h, 128
TY_BOOL_RGB_FLASHLIGHT	TY_COMPONENT_DEPTH_CAM
TYDefs.h, 130	TYDefs.h, 128
TY_BOOL_SGBM_HFILTER_HALF_WIN	TY_COMPONENT_DEVICE
TYDefs.h, 131	TYDefs.h, 128
TY_BOOL_SGBM_LRC	TY_COMPONENT_ID
TYDefs.h, 131	TYDefs.h, 124
TY BOOL SGBM MEDFILTER	TY COMPONENT IMU
TYDefs.h, 131	TYDefs.h, 128
TY_BOOL_SGPM_EPI_EN	TY_COMPONENT_IR_CAM_LEFT

TYDefs.h, 128	TY_GYRO_MISALIGNMENT, 23
TY_COMPONENT_IR_CAM_RIGHT	TYDefs.h, 126
TYDefs.h, 128	TY_GYRO_SCALE, 24
TY_COMPONENT_LASER	TYDefs.h, 127
TYDefs.h, 128	TY IMAGE DATA, 25
TY COMPONENT RGB CAM	TY_IMU_DATA, 25
TYDefs.h, 128	TY_INT_ANALOG_GAIN
TY_COMPONENT_RGB_CAM_LEFT	TYDefs.h, 130
TYDefs.h, 128	TY_INT_B_GAIN
TY COMPONENT RGB CAM RIGHT	TYDefs.h, 130
TYDefs.h, 128	TY_INT_CAPTURE_TIME_US
TY_COMPONENT_STORAGE	TYDefs.h, 130
TYDefs.h, 128	TY_INT_DEPTH_MAX_MM
TY_DECLARE_IMAGE_MODE1	TYDefs.h, 131
TYDefs.h, 121	TY_INT_DEPTH_MIN_MM
TY_DEVICE_BASE_INFO, 17	TYDefs.h, 131
TYDefs.h, 125	TY INT EXPOSURE TIME
TY_DEVICE_COMPONENT_LIST	TYDefs.h, 130
TYDefs.h, 125, 128	TY_INT_FILTER_THRESHOLD
TY_DEVICE_NET_INFO, 18	TYDefs.h, 131
TY_DEVICE_USB_INFO, 19	TY_INT_FRAME_PER_TRIGGER
TY DI WORKMODE, 19	TYDefs.h, 129
TY_DO_WORKMODE, 20	TY_INT_G_GAIN
TY_ENUM_DEPTH_QUALITY	TYDefs.h, 130
TYDefs.h, 131	TY_INT_GAIN
TY_ENUM_ENTRY, 20	TYDefs.h, 130
TYDefs.h, 125	TY_INT_HEIGHT
TY_ENUM_IMAGE_MODE	TYDefs.h, 129
TYDefs.h, 129	TY_INT_IR_FLASHLIGHT_INTENSITY
TY_ENUM_IMU_FPS	TYDefs.h, 130
TYDefs.h, 131	TY_INT_KEEP_ALIVE_TIMEOUT
TY_ENUM_STREAM_ASYNC	TYDefs.h, 130
TYDefs.h, 130	TY_INT_LASER_POWER
TY_ENUM_TIME_SYNC_TYPE	TYDefs.h, 130
TYDefs.h, 130	TY_INT_LINK_CMD_TIMEOUT
TY_ENUM_TRIGGER_POL	TYDefs.h, 129
TYDefs.h, 129	TY_INT_MAX_SPECKLE_DIFF
TY_ENUM_USER_SET_CURRENT	TYDefs.h, 132
TYDefs.h, 129	TY_INT_MAX_SPECKLE_SIZE
TY_ENUM_USER_SET_SELECTOR	TYDefs.h, 132
TYDefs.h, 129	TY_INT_NTP_SERVER_IP
TY_EVENT_INFO, 21	TYDefs.h, 129
TY_FEATURE_ID	TY_INT_PACKET_DELAY
TYDefs.h, 125	TYDefs.h, 129
TY_FEATURE_ID_LIST	TY_INT_R_GAIN
TYDefs.h, 129	TYDefs.h, 130
TY_FEATURE_INFO, 21	TY_INT_RANGE, 26
TY_FLOAT_EXPOSURE_TIME_US	TY_INT_RGB_FLASHLIGHT_INTENSITY
TYDefs.h, 130	TYDefs.h, 130
TY_FLOAT_RANGE, 22	TY_INT_SGBM_DISPARITY_NUM
TYDefs.h, 126	TYDefs.h, 131
TY_FLOAT_SCALE_UNIT	TY_INT_SGBM_DISPARITY_OFFSET
TYDefs.h, 129	TYDefs.h, 131
TY_FLOAT_SGPM_EPI_HS	TY_INT_SGBM_IMAGE_NUM
TYDefs.h, 131	TYDefs.h, 131
TY_FRAME_DATA, 22	TY_INT_SGBM_LRC_DIFF
TY_GYRO_BIAS, 23	TYDefs.h, 131
TYDefs.h, 126	TY_INT_SGBM_MATCH_WIN_HEIGHT

TYDefs.h, 131	TYDefs.h, 127
TY_INT_SGBM_MATCH_WIN_WIDTH	TY_INTERFACE_TYPE_LIST
TYDefs.h, 131	TYDefs.h, 127, 132
TY_INT_SGBM_MEDFILTER_THRESH	TY_LASER_PARAM, 27
TYDefs.h, 131	TY_LASER_PATTERN_PARAM, 27
TY_INT_SGBM_SEMI_PARAM_P1	TY_PHC_GROUP_ATTR, 28
TYDefs.h, 131	TY_PHC_GROUP_ATTR::phc_group_attr, 10
TY_INT_SGBM_SEMI_PARAM_P1_SCALE	TY_PIXEL_BITS_LIST
TYDefs.h, 131	TYDefs.h, 127, 132
TY_INT_SGBM_SEMI_PARAM_P2	TY_PIXEL_COLOR_DESC, 29
TYDefs.h, 131	TY_PIXEL_DESC, 29
TY_INT_SGBM_TEXTURE_OFFSET	TY_PIXEL_FORMAT_BAYER8BG
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGBM_TEXTURE_THRESH	TY_PIXEL_FORMAT_BAYER8GB
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGBM_UNIQUE_ABSDIFF	TY_PIXEL_FORMAT_BAYER8GR
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGBM_UNIQUE_FACTOR	TY_PIXEL_FORMAT_BAYER8RG TYDefs.h, 133
TYDefs.h, 131 TY_INT_SGBM_UNIQUE_MAX_COST	TY_PIXEL_FORMAT_BGR
TYDefs.h, 131	TYDefs.h, 133
TY INT SGPM EPI CH0	TY_PIXEL_FORMAT_BGR48
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGPM_EPI_CH1	
TYDefs.h, 131	TY_PIXEL_FORMAT_CSI_BAYER10BGGR TYDefs.h, 133
TY_INT_SGPM_EPI_HF	TY_PIXEL_FORMAT_CSI_BAYER10GBRG
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGPM_EPI_THRESH	TY_PIXEL_FORMAT_CSI_BAYER10GRBG
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGPM_NORMAL_PHASE_OFFSET	TY_PIXEL_FORMAT_CSI_BAYER10RGGB
TYDefs.h, 131	TYDefs.h, 133
TY_INT_SGPM_NORMAL_PHASE_SCALE	TY_PIXEL_FORMAT_CSI_BAYER12BGGR
TYDefs.h, 131	TYDefs.h, 133
TY INT SGPM ORDER FILTER CHN	TY_PIXEL_FORMAT_CSI_BAYER12GBRG
TYDefs.h, 131	TYDefs.h, 133
TY INT SGPM PHASE NUM	TY_PIXEL_FORMAT_CSI_BAYER12GRBG
TYDefs.h, 131	TYDefs.h, 133
TY INT SGPM REF PHASE OFFSET	TY_PIXEL_FORMAT_CSI_BAYER12RGGB
TYDefs.h, 131	TYDefs.h, 133
TY INT SGPM REF PHASE SCALE	TY_PIXEL_FORMAT_CSI_MONO10
TYDefs.h, 131	TYDefs.h, 133
TY_INT_TOF_ANTI_SUNLIGHT_INDEX	TY_PIXEL_FORMAT_CSI_MONO12
TYDefs.h, 132	TYDefs.h, 133
TY_INT_TOF_CHANNEL	TY PIXEL FORMAT DEPTH16
TYDefs.h, 131	TYDefs.h, 133
TY_INT_TOF_HDR_RATIO	TY_PIXEL_FORMAT_JPEG
TYDefs.h, 130	TYDefs.h, 133
TY_INT_TOF_JITTER_THRESHOLD	TY_PIXEL_FORMAT_LIST
TYDefs.h, 130	TYDefs.h, 132
TY_INT_TOF_MODULATION_THRESHOLD	TY PIXEL FORMAT MJPG
TYDefs.h, 131	TYDefs.h, 133
TY_INT_TRIGGER_DELAY_US	TY_PIXEL_FORMAT_MONO
TYDefs.h, 130	TYDefs.h, 133
TY_INT_TRIGGER_DURATION_US	TY_PIXEL_FORMAT_MONO16
TYDefs.h, 130	TYDefs.h, 133
TY_INT_WIDTH	TY_PIXEL_FORMAT_RGB
TYDefs.h, 129	TYDefs.h, 133
TY_INTERFACE_INFO, 26	TY_PIXEL_FORMAT_RGB48

TYDefs.h, 133	TYDefs.h, 130
TY_PIXEL_FORMAT_TOF_IR_MONO16	TY_STRUCT_CAM_CALIB_DATA
TYDefs.h, 133	TYDefs.h, 129
TY_PIXEL_FORMAT_XYZ48	TY_STRUCT_CAM_DISTORTION
TYDefs.h, 133	TYDefs.h, 129
TY_PIXEL_FORMAT_YUYV	TY_STRUCT_CAM_INTRINSIC
TYDefs.h, 133	TYDefs.h, 129
TY_PIXEL_FORMAT_YVYU	TY_STRUCT_CAM_RECTIFIED_INTRI
TYDefs.h, 133	TYDefs.h, 129
TY_RESOLUTION_MODE_1280x720	TY_STRUCT_CAM_RECTIFIED_ROTATION
TYDefs.h, 134	TYDefs.h, 129
TY_RESOLUTION_MODE_1280x800	TY_STRUCT_CAM_STATISTICS
TYDefs.h, 134	TYDefs.h, 129
TY_RESOLUTION_MODE_1280x960	TY_STRUCT_DI0_WORKMODE
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_1600x1200	TY_STRUCT_DI1_WORKMODE
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_160x100 TYDefs.h, 133	TY_STRUCT_DI2_WORKMODE
TY_RESOLUTION_MODE_160x120	TYDefs.h, 130 TY_STRUCT_DO0_WORKMODE
TYDefs.h, 133	TYDefs.h, 130
TY_RESOLUTION_MODE_1920x1080	TY_STRUCT_DO1_WORKMODE
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_1920x1440	TY_STRUCT_DO2_WORKMODE
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_2048x1536	TY_STRUCT_EXTRINSIC_TO_DEPTH
TYDefs.h, 134	TYDefs.h, 129
TY_RESOLUTION_MODE_240x320	TY_STRUCT_EXTRINSIC_TO_IR_LEFT
TYDefs.h, 133	TYDefs.h, 129
TY_RESOLUTION_MODE_240x96	TY_STRUCT_FLOOD_ENABLE_BY_IDX
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_2560x1920	TY_STRUCT_FLOOD_POWER_BY_IDX
TYDefs.h, 134	TYDefs.h, 130
TY RESOLUTION MODE 2592x1944	TY_STRUCT_IMU_ACC_BIAS
TYDefs.h, 134	TYDefs.h, 131
TY_RESOLUTION_MODE_320x180	TY_STRUCT_IMU_ACC_MISALIGNMENT
TYDefs.h, 133	TYDefs.h, 131
TY_RESOLUTION_MODE_320x200	TY_STRUCT_IMU_ACC_SCALE
TYDefs.h, 133	TYDefs.h, 131
TY_RESOLUTION_MODE_320x240	TY STRUCT IMU CAM TO IMU
TYDefs.h, 133	TYDefs.h, 131
TY_RESOLUTION_MODE_480x640	TY STRUCT IMU GYRO BIAS
TYDefs.h, 133	TYDefs.h, 131
TY RESOLUTION MODE 640x360	TY_STRUCT_IMU_GYRO_MISALIGNMENT
TYDefs.h, 133	TYDefs.h, 131
TY_RESOLUTION_MODE_640x400	TY_STRUCT_IMU_GYRO_SCALE
TYDefs.h, 134	TYDefs.h, 131
TY_RESOLUTION_MODE_640x480	TY_STRUCT_LASER_ENABLE_BY_IDX
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_800x600	TY_STRUCT_LASER_POWER_BY_IDX
TYDefs.h, 134	TYDefs.h, 130
TY_RESOLUTION_MODE_960x1280	TY_STRUCT_PHC_GROUP_ATTR
TYDefs.h, 134	TYDefs.h, 131
TY_RESOLUTION_MODE_LIST	TY_STRUCT_TOF_FREQ
TYDefs.h, 133	TYDefs.h, 132
TY_STRING_USER_SET_DESCRIPTION	TY_STRUCT_TRIGGER_PARAM
TYDefs.h, 129	TYDefs.h, 129
TY_STRUCT_AEC_ROI	TY_STRUCT_TRIGGER_PARAM_EX

TYDefs.h, 129	TYGetEnumEntryInfo, 65
TY_STRUCT_TRIGGER_TIMER_LIST	TYGetFeatureInfo, 66
TYDefs.h, 129	TYGetFloat, 67
TY_STRUCT_TRIGGER_TIMER_PERIOD	TYGetFloatRange, 69
TYDefs.h, 130	TYGetFrameBufferSize, 70
TY_TEMP_DATA, 30	TYGetInt, 71
TY_TOF_FREQ, 30	TYGetInterfaceList, 72
TY TRIGGER MODE LIST	TYGetInterfaceNumber, 73
TYDefs.h, 127, 134	TYGetIntRange, 74
	_
TY_TRIGGER_MODE_M_PER	TYGetString, 75
TYDefs.h, 134	TYGetStringLength, 76
TY_TRIGGER_MODE_M_SIG	TYGetStruct, 78
TYDefs.h, 134	TYHasDevice, 80
TY_TRIGGER_MODE_OFF	TYHasFeature, 81
TYDefs.h, 134	TYHasInterface, 82
TY_TRIGGER_MODE_PER_PASS	TYLibVersion, 82
TYDefs.h, 134	TYOpenDevice, 83
TY_TRIGGER_MODE_PER_PASS2	TYOpenDeviceWithIP, 84
TYDefs.h, 134	TYOpenInterface, 86
TY_TRIGGER_MODE_SIG_PASS	TYRegisterEventCallback, 86
TYDefs.h, 134	TYRegisterImuCallback, 87
TY_TRIGGER_MODE_SLAVE	TYRemoveLogFile, 88
TYDefs.h, 134	TYRemoveLogServer, 88
TY TRIGGER PARAM, 30	TYSendSoftTrigger, 89
TY TRIGGER PARAM EX, 31	TYSetBool, 90
TY_TRIGGER_TIMER_LIST, 31	TYSetByteArray, 91
TY_TRIGGER_TIMER_PERIOD, 32	TYSetEnum, 93
TY_VECT_3F, 32	TYSetFloat, 94
TY_VERSION_INFO, 32	TYSetInt, 96
TYApi.h, 35	TYSetLogLevel, 97
TYAppendLogToFile, 39	TYSetLogPrefix, 97
TYAppendLogToServer, 40	TYSetLogFrenx, 97 TYSetString, 98
	-
TYClear Burian 41	TYSetStruct, 99
TYCloseDevice, 41	TYStartCapture, 101
TYCloseInterface, 42	TYStopCapture, 102
TYDeinitLib, 42	TYUpdateAllDeviceList, 103
TYDisableComponents, 43	TYUpdateDeviceList, 103
TYEnableComponents, 44	TYUpdateInterfaceList, 104
TYEnqueueBuffer, 45	TYAppendLogToFile
TYErrorString, 46	TYApi.h, 39
TYFetchFrame, 46	TYAppendLogToServer
TYForceDeviceIP, 47	TYApi.h, 40
TYGetBool, 48	TYClearBufferQueue
TYGetByteArray, 50	TYApi.h, 41
TYGetByteArrayAttr, 51	TYCloseDevice
TYGetByteArraySize, 53	TYApi.h, 41
TYGetComponentIDs, 54	TYCloseInterface
TYGetDeviceFeatureInfo, 55	TYApi.h, 42
TYGetDeviceFeatureNumber, 56	TYCoordinateMapper.h, 104
TYGetDeviceInfo, 57	TYDepthImageFillEmptyRegion, 107
TYGetDeviceInterface, 58	TYInvertExtrinsic, 107
TYGetDeviceList, 59	TYMAP_CHECKRET, 106
TYGetDeviceNumber, 59	TYMapDepthImageToPoint3d, 108
TYGetDeviceXML, 60	TYMapDepthToPoint3d, 108
TYGetDeviceXMLSize, 61	TYMapPoint3dToDepth, 109
TYGetEnabledComponents, 62	TYMapPoint3dToDepthImage, 109
TYGetEnum, 63	TYMapPoint3dToPoint3d, 110
TYGetEnumEntryCount, 64	TYDefs.h, 110
i i detenumenti yoount, 04	11000.11, 110

TY_ACC_BIAS, 121	TY_ENUM_TRIGGER_POL, 129
TY_ACC_MISALIGNMENT, 121	TY_ENUM_USER_SET_CURRENT, 129
TY_ACC_SCALE, 122	TY_ENUM_USER_SET_SELECTOR, 129
TY_ACCESS_MODE_LIST, 122, 128	TY_FEATURE_ID, 125
TY_BOOL_AUTO_AWB, 130	TY_FEATURE_ID_LIST, 129
TY_BOOL_AUTO_EXPOSURE, 130	TY_FLOAT_EXPOSURE_TIME_US, 130
TY_BOOL_AUTO_GAIN, 130	TY_FLOAT_RANGE, 126
TY_BOOL_BRIGHTNESS_HISTOGRAM, 130	TY_FLOAT_SCALE_UNIT, 129
TY BOOL CMOS SYNC, 130	TY FLOAT SGPM EPI HS, 131
TY_BOOL_DEPTH_POSTPROC, 130	TY GYRO BIAS, 126
TY_BOOL_HDR, 130	TY_GYRO_MISALIGNMENT, 126
TY_BOOL_IMU_DATA_ONOFF, 130	TY_GYRO_SCALE, 127
TY_BOOL_IR_FLASHLIGHT, 130	TY_INT_ANALOG_GAIN, 130
TY_BOOL_KEEP_ALIVE_ONOFF, 130	TY_INT_B_GAIN, 130
TY_BOOL_LASER_AUTO_CTRL, 130	TY_INT_CAPTURE_TIME_US, 130
TY_BOOL_RGB_FLASHLIGHT, 130	TY_INT_DEPTH_MAX_MM, 131
TY_BOOL_SGBM_HFILTER_HALF_WIN, 131	TY_INT_DEPTH_MIN_MM, 131
TY_BOOL_SGBM_LRC, 131	TY_INT_EXPOSURE_TIME, 130
TY_BOOL_SGBM_MEDFILTER, 131	TY_INT_FILTER_THRESHOLD, 131
TY_BOOL_SGPM_EPI_EN, 131	TY_INT_FRAME_PER_TRIGGER, 129
TY_BOOL_SGPM_ORDER_FILTER_EN, 131	TY_INT_G_GAIN, 130
TY_BOOL_TIME_SYNC_READY, 130	TY_INT_GAIN, 130
TY_BOOL_TOF_ANTI_INTERFERENCE, 132	TY_INT_HEIGHT, 129
TY_BOOL_TRIGGER_OUT_IO, 130	TY_INT_IR_FLASHLIGHT_INTENSITY, 130
TY_BOOL_UNDISTORTION, 130	TY_INT_KEEP_ALIVE_TIMEOUT, 130
TY_BYTEARRAY_ATTR, 122	TY_INT_LASER_POWER, 130
TY_BYTEARRAY_CUSTOM_BLOCK, 129	TY_INT_LINK_CMD_TIMEOUT, 129
TY_BYTEARRAY_HDR_PARAMETER, 130	TY_INT_MAX_SPECKLE_DIFF, 132
TY_BYTEARRAY_ISP_BLOCK, 129	TY_INT_MAX_SPECKLE_SIZE, 132
TY_CAMERA_CALIB_INFO, 122	TY_INT_NTP_SERVER_IP, 129
TY_CAMERA_DISTORTION, 123	TY_INT_PACKET_DELAY, 129
TY_CAMERA_EXTRINSIC, 123	TY_INT_R_GAIN, 130
TY_CAMERA_INTRINSIC, 123	TY_INT_RGB_FLASHLIGHT_INTENSITY, 130
TY_CAMERA_ROTATION, 124	TY_INT_SGBM_DISPARITY_NUM, 131
TY_CAMERA_TO_IMU, 124	TY_INT_SGBM_DISPARITY_OFFSET, 131
TY_CMD_USER_SET_LOAD, 129	TY_INT_SGBM_IMAGE_NUM, 131
TY_CMD_USER_SET_SAVE, 129	TY_INT_SGBM_LRC_DIFF, 131
TY_COMPONENT_BRIGHT_HISTO, 128	TY_INT_SGBM_MATCH_WIN_HEIGHT, 131
TY_COMPONENT_DEPTH_CAM, 128	TY_INT_SGBM_MATCH_WIN_WIDTH, 131
TY_COMPONENT_DEVICE, 128	TY_INT_SGBM_MEDFILTER_THRESH, 131
TY_COMPONENT_ID, 124	TY_INT_SGBM_SEMI_PARAM_P1, 131
TY_COMPONENT_IMU, 128	TY_INT_SGBM_SEMI_PARAM_P1_SCALE, 131
TY_COMPONENT_IR_CAM_LEFT, 128	TY_INT_SGBM_SEMI_PARAM_P2, 131
TY_COMPONENT_IR_CAM_RIGHT, 128	TY_INT_SGBM_TEXTURE_OFFSET, 131
TY_COMPONENT_LASER, 128	TY INT SGBM TEXTURE THRESH, 131
TY_COMPONENT_RGB_CAM, 128	TY_INT_SGBM_UNIQUE_ABSDIFF, 131
TY_COMPONENT_RGB_CAM_LEFT, 128	TY_INT_SGBM_UNIQUE_FACTOR, 131
TY_COMPONENT_RGB_CAM_RIGHT, 128	TY_INT_SGBM_UNIQUE_MAX_COST, 131
TY_COMPONENT_STORAGE, 128	TY_INT_SGPM_EPI_CH0, 131
TY_DECLARE_IMAGE_MODE1, 121	TY_INT_SGPM_EPI_CH1, 131
TY_DEVICE_BASE_INFO, 125	TY_INT_SGPM_EPI_HF, 131
TY_DEVICE_COMPONENT_LIST, 125, 128	TY_INT_SGPM_EPI_THRESH, 131
TY_ENUM_DEPTH_QUALITY, 131	TY_INT_SGPM_NORMAL_PHASE_OFFSET, 131
TY_ENUM_ENTRY, 125	TY_INT_SGPM_NORMAL_PHASE_SCALE, 131
TY_ENUM_IMAGE_MODE, 129	TY_INT_SGPM_ORDER_FILTER_CHN, 131
TY_ENUM_IMU_FPS, 131	TY_INT_SGPM_PHASE_NUM, 131
TY_ENUM_STREAM_ASYNC, 130	TY_INT_SGPM_REF_PHASE_OFFSET, 131
TY_ENUM_TIME_SYNC_TYPE, 130	TY_INT_SGPM_REF_PHASE_SCALE, 131

TY_INT_TOF_ANTI_SUNLIGHT_INDEX, 132	TY_RESOLUTION_MODE_640x480, 134
TY_INT_TOF_CHANNEL, 131	TY RESOLUTION MODE 800x600, 134
TY_INT_TOF_HDR_RATIO, 130	TY_RESOLUTION_MODE_960x1280, 134
TY_INT_TOF_JITTER_THRESHOLD, 130	TY_RESOLUTION_MODE_LIST, 133
TY_INT_TOF_MODULATION_THRESHOLD, 131	TY_STRING_USER_SET_DESCRIPTION, 129
TY_INT_TRIGGER_DELAY_US, 130	TY_STRUCT_AEC_ROI, 130
TY_INT_TRIGGER_DURATION_US, 130	TY_STRUCT_CAM_CALIB_DATA, 129
TY_INT_WIDTH, 129	TY_STRUCT_CAM_DISTORTION, 129
TY_INTERFACE_INFO, 127	TY_STRUCT_CAM_INTRINSIC, 129
TY_INTERFACE_TYPE_LIST, 127, 132	TY_STRUCT_CAM_RECTIFIED_INTRI, 129
TY_PIXEL_BITS_LIST, 127, 132	TY_STRUCT_CAM_RECTIFIED_ROTATION, 129
TY_PIXEL_FORMAT_BAYER8BG, 133	TY_STRUCT_CAM_STATISTICS, 129
TY_PIXEL_FORMAT_BAYER8GB, 133	TY_STRUCT_DI0_WORKMODE, 130
TY_PIXEL_FORMAT_BAYER8GR, 133	TY STRUCT DI1 WORKMODE, 130
TY_PIXEL_FORMAT_BAYER8RG, 133	TY STRUCT DI2 WORKMODE, 130
TY_PIXEL_FORMAT_BGR, 133	TY_STRUCT_DO0_WORKMODE, 130
TY_PIXEL_FORMAT_BGR48, 133	TY STRUCT DO1 WORKMODE, 130
TY_PIXEL_FORMAT_CSI_BAYER10BGGR, 133	TY STRUCT DO2 WORKMODE, 130
TY_PIXEL_FORMAT_CSI_BAYER10GBRG, 133	TY_STRUCT_EXTRINSIC_TO_DEPTH, 129
TY_PIXEL_FORMAT_CSI_BAYER10GRBG, 133	TY_STRUCT_EXTRINSIC_TO_IR_LEFT, 129
TY_PIXEL_FORMAT_CSI_BAYER10RGGB, 133	TY_STRUCT_FLOOD_ENABLE_BY_IDX, 130
TY_PIXEL_FORMAT_CSI_BAYER12BGGR, 133	TY_STRUCT_FLOOD_POWER_BY_IDX, 130
TY_PIXEL_FORMAT_CSI_BAYER12GBRG, 133	TY_STRUCT_IMU_ACC_BIAS, 131
TY_PIXEL_FORMAT_CSI_BAYER12GRBG, 133	TY_STRUCT_IMU_ACC_MISALIGNMENT, 131
TY_PIXEL_FORMAT_CSI_BAYER12RGGB, 133	TY_STRUCT_IMU_ACC_SCALE, 131
TY_PIXEL_FORMAT_CSI_MONO10, 133	TY_STRUCT_IMU_CAM_TO_IMU, 131
TY_PIXEL_FORMAT_CSI_MONO12, 133	TY_STRUCT_IMU_GYRO_BIAS, 131
TY_PIXEL_FORMAT_DEPTH16, 133	TY_STRUCT_IMU_GYRO_MISALIGNMENT, 131
TY_PIXEL_FORMAT_JPEG, 133	TY_STRUCT_IMU_GYRO_SCALE, 131
TY_PIXEL_FORMAT_LIST, 132	TY_STRUCT_LASER_ENABLE_BY_IDX, 130
TY_PIXEL_FORMAT_MJPG, 133	TY_STRUCT_LASER_POWER_BY_IDX, 130
TY_PIXEL_FORMAT_MONO, 133	TY_STRUCT_PHC_GROUP_ATTR, 131
TY_PIXEL_FORMAT_MONO16, 133	TY_STRUCT_TOF_FREQ, 132
TY_PIXEL_FORMAT_RGB, 133	TY_STRUCT_TRIGGER_PARAM, 129
TY_PIXEL_FORMAT_RGB48, 133	TY STRUCT TRIGGER PARAM EX, 129
TY_PIXEL_FORMAT_TOF_IR_MONO16, 133	TY_STRUCT_TRIGGER_TIMER_LIST, 129
TY_PIXEL_FORMAT_XYZ48, 133	TY_STRUCT_TRIGGER_TIMER_PERIOD, 130
TY_PIXEL_FORMAT_YUYV, 133	TY_TRIGGER_MODE_LIST, 127, 134
TY_PIXEL_FORMAT_YVYU, 133	TY_TRIGGER_MODE_M_PER, 134
TY_RESOLUTION_MODE_1280x720, 134	TY_TRIGGER_MODE_M_SIG, 134
TY_RESOLUTION_MODE_1280x800, 134	TY_TRIGGER_MODE_OFF, 134
TY_RESOLUTION_MODE_1280x960, 134	TY_TRIGGER_MODE_PER_PASS, 134
TY_RESOLUTION_MODE_1600x1200, 134	TY_TRIGGER_MODE_PER_PASS2, 134
TY_RESOLUTION_MODE_160x100, 133	TY_TRIGGER_MODE_SIG_PASS, 134
TY_RESOLUTION_MODE_160x120, 133	TY_TRIGGER_MODE_SLAVE, 134
TY_RESOLUTION_MODE_1920x1080, 134	TYDeinitLib
TY_RESOLUTION_MODE_1920x1440, 134	TYApi.h, 42
TY_RESOLUTION_MODE_2048x1536, 134	TYDepthEnhenceFilter
TY_RESOLUTION_MODE_240x320, 133	TYImageProc.h, 136
TY RESOLUTION MODE 240x96, 134	TYDepthImageFillEmptyRegion
TY_RESOLUTION_MODE_2560x1920, 134	TYCoordinateMapper.h, 107
TY_RESOLUTION_MODE_2592x1944, 134	TYDepthSpeckleFilter
TY_RESOLUTION_MODE_320x180, 133	TYImageProc.h, 136
TY_RESOLUTION_MODE_320x200, 133	TYDisableComponents
TY_RESOLUTION_MODE_320x240, 133	TYApi.h, 43
	TYEnableComponents
TY_RESOLUTION_MODE_480x640, 133	•
TY_RESOLUTION_MODE_640x360, 133	TYApi.h, 44
TY_RESOLUTION_MODE_640x400, 134	TYEnqueueBuffer

TYApi.h, 45	TYGetString
TYEnumEntry, 33	TYApi.h, 75
TYErrorString	TYGetStringLength
TYApi.h, 46	TYApi.h, 76
TYFetchFrame	TYGetStruct
TYApi.h, 46	TYApi.h, 78
TYForceDeviceIP	TYHasDevice
TYApi.h, 47	TYApi.h, 80
TYGetBool	TYHasFeature
TYApi.h, 48	TYApi.h, 81
TYGetByteArray	TYHasInterface
TYApi.h, 50	TYApi.h, 82
TYGetByteArrayAttr	TYImageProc.h, 134
TYApi.h, 51	TYDepthEnhenceFilter, 136
TYGetByteArraySize	TYDepthSpeckleFilter, 136
TYApi.h, 53	TYImageProcesAcceEnable, 137
TYGetComponentIDs	TYUndistortImage, 137
TYApi.h, 54	TYImageProcesAcceEnable
TYGetDeviceFeatureInfo	TYImageProc.h, 137
TYApi.h, 55	TYInvertExtrinsic
TYGetDeviceFeatureNumber	TYCoordinateMapper.h, 107
TYApi.h, 56	TYLibVersion
TYGetDeviceInfo	TYApi.h, 82
TYApi.h, 57	TYMAP CHECKRET
TYGetDeviceInterface	TYCoordinateMapper.h, 106
TYApi.h, 58	TYMapDepthImageToPoint3d
TYGetDeviceList	TYCoordinateMapper.h, 108
TYApi.h, 59	TYMapDepthToPoint3d
TYGetDeviceNumber	TYCoordinateMapper.h, 108
TYApi.h, 59	TYMapPoint3dToDepth
TYGetDeviceXML	TYCoordinateMapper.h, 109
TYApi.h, 60	TYMapPoint3dToDepthImage
TYGetDeviceXMLSize	
	TYCoordinateMapper.h, 109
TYApi.h, 61	TYMapPoint3dToPoint3d
TYGetEnabledComponents	TYCoordinateMapper.h, 110
TYApi.h, 62	TYOpenDevice
TYGetEnum	TYApi.h, 83
TYApi.h, 63	TYOpenDeviceWithIP
TYGetEnumEntryCount	TYApi.h, 84
TYApi.h, 64	TYOpenInterface
TYGetEnumEntryInfo	TYApi.h, 86
TYApi.h, 65	TYRegisterEventCallback
TYGetFeatureInfo	TYApi.h, 86
TYApi.h, 66	TYRegisterImuCallback
TYGetFloat	TYApi.h, 87
TYApi.h, 67	TYRemoveLogFile
TYGetFloatRange	TYApi.h, 88
TYApi.h, 69	TYRemoveLogServer
TYGetFrameBufferSize	TYApi.h, 88
TYApi.h, 70	TYSendSoftTrigger
TYGetInt	TYApi.h, 89
TYApi.h, 71	TYSetBool
TYGetInterfaceList	TYApi.h, 90
TYApi.h, 72	TYSetByteArray
TYGetInterfaceNumber	TYApi.h, 91
TYApi.h, 73	TYSetEnum
TYGetIntRange	TYApi.h, 93
TYApi.h, 74	TYSetFloat
, ,,,,,,	

```
TYApi.h, 94
TYSetInt
    TYApi.h, 96
TYSetLogLevel
    TYApi.h, 97
TYSetLogPrefix
    TYApi.h, 97
TYSetString
    TYApi.h, 98
TYSetStruct
    TYApi.h, 99
TYStartCapture
    TYApi.h, 101
TYStopCapture
    TYApi.h, 102
TYUndistortImage
    TYImageProc.h, 137
TYUpdateAllDeviceList
    TYApi.h, 103
TYUpdateDeviceList
    TYApi.h, 103
TYUpdateInterfaceList
    TYApi.h, 104
unit_size
    TY_BYTEARRAY_ATTR, 12
valid_size
    TY_BYTEARRAY_ATTR, 13
z_stream_s, 33
```