Camera SDK Features Guide

C#

Release Date: June 11, 2025

Version: V1.0

SHANGHAI PERCIPIO TECHNOLOGY LIMITED

Table of Contents

1 Features Under the *Device* Component

1.1 TY_BOOL_GVSP_RESEND

```
DevParam param = cl.DevParamFromBool(false);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_GVSP_RESEND, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_GVSP_RESEND);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

1.2 TY_INT_ACCEPTABLE_PERCENT

```
DevParam param = cl.DevParamFromInt(90);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_ACCEPTABLE_PERCENT, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE,

TY_INT_ACCEPTABLE_PERCENT);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

1.3 TY_INT_NTP_SERVER_IP

```
int ip = cl.IPv4StringToInt("0.0.0.0");

DevParam param = cl.DevParamFromInt(ip);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_NTP_SERVER_IP, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_NTP_SERVER_IP);

int read_param_m = read_param.toInt();

Console.WriteLine($"{read_param_m}");
```

1.4 TY_INT_PACKET_SIZE

```
DevParam param = cl.DevParamFromInt(1500);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PACKET_SIZE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PACKET_SIZE);
```

```
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

1.5 TY_ENUM_TRIGGER_POL

```
DevParam param = cl.DevParamFromEnum(TY_TRIGGER_POL_FALLINGEDGE);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TRIGGER_POL, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TRIGGER_POL);
uint m_read_param = read_param.toEnum();
Console.WriteLine($"current value {m_read_param}");
EnumEntryVector m_read_param2 = read_param.eList();
for (int i = 0; i < m_read_param2.Count(); i++)
{
Console.WriteLine($"{ m_read_param2[i].value}");
}
```

1.6 TY_INT_FRAME_PER_TRIGGER

```
DevParam param = cl.DevParamFromInt(5);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_FRAME_PER_TRIGGER, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_FRAME_PER_TRIGGER);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
EnumEntryVector m_read_param2 = read_param.eList();
for (int i = 0; i < m_read_param2.Count(); i++)
{
Console.WriteLine($"{ m_read_param2[i].value}");
}
```

1.7 TY_BOOL_KEEP_ALIVE_ONOFF

```
DevParam param = cl.DevParamFromBool(false);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_KEEP_ALIVE_ONOFF, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_KEEP_ALIVE_ONOFF);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

1.8 TY_INT_KEEP_ALIVE_TIMEOUT

```
DevParam param = cl.DevParamFromInt(3000);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_KEEP_ALIVE_TIMEOUT, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_KEEP_ALIVE_TIMEOUT);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

1.9 TY_INT_PACKET_DELAY

```
DevParam param = cl.DevParamFromInt(0);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PACKET_DELAY, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PACKET_DELAY);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

1.10 TY_BOOL_CMOS_SYNC

```
DevParam param = cl.DevParamFromBool(false);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_CMOS_SYNC, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_CMOS_SYNC);
```

```
bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

1.11 TY_ENUM_STREAM_ASYNC

```
DevParam param = cl.DevParamFromEnumt(TY_STREAM_ASYNC_ALL);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_STREAM_ASYNC, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_STREAM_ASYNC);

uint m_read_param = read_param.toEnum();

Console.WriteLine($"current value {m_read_param}");

EnumEntryVector m_read_param2 = read_param.eList();

for (int i = 0; i < m_read_param2.Count(); i++)

{

Console.WriteLine($"{ m_read_param2[i].value}");

}
```

1.12 TY_INT_CAPTURE_TIME_US

```
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_CAPTURE_TIME_US);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");
```

1.13 TY_ENUM_TIME_SYNC_TYPE

```
DevParam param = cl.DevParamFromEnum(TY_TIME_SYNC_TYPE_HOST);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TIME_SYNC_TYPE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TIME_SYNC_TYPE);

uint m_read_param = read_param.toEnum();

Console.WriteLine($"current value {m_read_param}");

EnumEntryVector m_read_param2 = read_param.eList();

for (int i = 0; i < m_read_param2.Count(); i++){

Console.WriteLine($"{ m_read_param2[i].value}");

}
```

1.14 TY_BOOL_TIME_SYNC_READY

```
DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_BOOL_TIME_SYNC_READY);

bool m_status = status.toBool();
```

```
Console.WriteLine($"current value {m_status}");
```

1.15 TY_INT_PERSISTENT_IP

```
int ip = cl.IPv4StringToInt("0.0.0.0");

DevParam param = cl.DevParamFromInt(ip);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PERSISTENT_IP, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PERSISTENT_IP);

int read_param_m = read_param.toInt();

Console.WriteLine($"{read_param_m}");
```

1.16 TY_INT_PERSISTENT_SUBMASK

```
int netmask = cl.IPv4StringToInt("0.0.0.0");

DevParam param1 = cl.DevParamFromInt(netmask);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PERSISTENT_SUBMASK, param1);

DevParam read_param1 = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE,

TY_INT_PERSISTENT_SUBMASK);

int read_param_m1 = read_param1.toInt();

Console.WriteLine($"{read_param_m1}");
```

1.17 TY_INT_PERSISTENT_GATEWAY

```
int gateway = cl.IPv4StringToInt("0.0.0.0");

DevParam param2 = cl.DevParamFromInt(gateway);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_INT_PERSISTENT_GATEWAY, param2);

DevParam read_param2 = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE,

TY_INT_PERSISTENT_GATEWAY);

int read_param_m2 = read_param2.toInt();

Console.WriteLine($"{read_param_m2}");
```

1.18 TY_ENUM_CONFIG_MODE

```
DevParam param = cl.DevParamFromEnum(TY_CONFIG_MODE_PRESET0);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_CONFIG_MODE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_CONFIG_MODE);

uint m_read_param = read_param.toEnum();

Console.WriteLine($"current value {m_read_param}");
```

```
EnumEntryVector m_read_param2 = read_param.eList();

for (int i = 0; i < m_read_param2.Count(); i++)

{

Console.WriteLine($"{ m_read_param2[i].value}");

}
```

1.19 TY_ENUM_TEMPERATURE_ID

```
DevParam param5 = cl.DevParamFromEnum(TY_TEMPERATURE_LEFT);

 int error5 = cl.DeviceSetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TEMPERATURE_ID, param5);

 DevParam read_param5 = cl.DeviceGetParameter(handle, TY_COMPONENT_DEVICE, TY_ENUM_TEMPERATURE_ID);

 uint m_read_param5 = read_param5.toEnum();

 Console.WriteLine($"current value {m_read_param5}");

 EnumEntryVector l_read_param5 = read_param5.eList();

 for (int i = 0; i < l_read_param5.Count(); i++)

 {

     float tmp= cl.DeviceControlReadTemperature(handle, l_read_param5[i].value);

     Console.WriteLine(string.Format("{0}={1}",l_read_param5[i].description,tmp)); }
```

2 Features Under the *Laser* Component

2.1 TY_BOOL_LASER_AUTO_CTRL

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_LASER_AUTO_CTRL, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_LASER_AUTO_CTRL);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

2.2 TY_INT_LASER_POWER

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_INT_LASER_POWER, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER, TY_INT_LASER_POWER);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

2.3 TY_BOOL_IR_FLASHLIGHT

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_IR_FLASHLIGHT, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_IR_FLASHLIGHT);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

2.4 TY_BOOL_RGB_FLASHLIGHT

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_RGB_FLASHLIGHT, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER, TY_BOOL_RGB_FLASHLIGHT);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

2.5 TY_INT_IR_FLASHLIGHT_INTENSITY

```
DevParam param = cl.DevParamFromInt(0);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_INT_IR_FLASHLIGHT_INTENSITY, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER,

TY_INT_IR_FLASHLIGHT_INTENSITY);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

2.6 TY_INT_RGB_FLASHLIGHT_INTENSITY

```
DevParam param = cl.DevParamFromInt(0);

cl.DeviceSetParameter(handle, TY_COMPONENT_LASER, TY_INT_RGB_FLASHLIGHT_INTENSITY, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_LASER,

TY_INT_RGB_FLASHLIGHT_INTENSITY);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3 Features Under the *Depth* Component

3.1 TY_FLOAT_SCALE_UNIT

```
DevParam param = cl.DevParamFromFloat(0.0125f);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_FLOAT_SCALE_UNIT, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_FLOAT_SCALE_UNIT);

float m_read_param = read_param.toFloat();

Console.WriteLine($"current value {m_read_param}");

float min = read_param.fMin();

float max = read_param.fMax();

float inc = read_param.fInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.2 TY_INT_TOF_HDR_RATIO

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_HDR_RATIO, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_HDR_RATIO);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

nt min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.3 TY_INT_TOF_JITTER_THRESHOLD

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_JITTER_THRESHOLD, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_TOF_JITTER_THRESHOLD);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();
```

```
int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.4 TY_INT_SGBM_IMAGE_NUM

```
DevParam param = cl.DevParamFromInt(3);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_IMAGE_NUM, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_IMAGE_NUM);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.5 TY_INT_SGBM_DISPARITY_NUM

```
DevParam param = cl.DevParamFromInt(3);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_DISPARITY_NUM, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_DISPARITY_NUM);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.6 TY_INT_SGBM_DISPARITY_OFFSET

```
DevParam param = cl.DevParamFromInt(3);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_DISPARITY_OFFSET, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_DISPARITY_OFFSET);

int m_read_param = read_param.toInt();
```

```
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.7 TY_INT_SGBM_MATCH_WIN_HEIGHT

```
DevParam param = cl.DevParamFromInt(3);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_MATCH_WIN_HEIGHT, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
TY_INT_SGBM_MATCH_WIN_HEIGHT);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.8 TY_INT_SGBM_SEMI_PARAM_P1

```
DevParam param = cl.DevParamFromInt(10000);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_SEMI_PARAM_P1, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
TY_INT_SGBM_SEMI_PARAM_P1);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.9 TY_INT_SGBM_SEMI_PARAM_P2

```
DevParam param = cl.DevParamFromInt(0);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_SEMI_PARAM_P2, param);
```

```
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
TY_INT_SGBM_SEMI_PARAM_P2);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.10 TY_INT_SGBM_UNIQUE_FACTOR

```
DevParam param = cl.DevParamFromInt(511);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_UNIQUE_FACTOR, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
TY_INT_SGBM_UNIQUE_FACTOR);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.11 TY_INT_SGBM_UNIQUE_ABSDIFF

```
DevParam param = cl.DevParamFromInt(10000);
cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_UNIQUE_ABSDIFF, param);
DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
TY_INT_SGBM_UNIQUE_ABSDIFF);
int m_read_param = read_param.toInt();
Console.WriteLine($"current value {m_read_param}");
int min = read_param.mMin();
int max = read_param.mMax();
int inc = read_param.mInc();
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.12 TY_BOOL_SGBM_HFILTER_HALF_WIN

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_SGBM_HFILTER_HALF_WIN, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_BOOL_SGBM_HFILTER_HALF_WIN);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

3.13 TY_INT_SGBM_MATCH_WIN_WIDTH

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_MATCH_WIN_WIDTH, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_MATCH_WIN_WIDTH);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.14 TY_BOOL_SGBM_MEDFILTER

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_SGBM_MEDFILTER, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_SGBM_MEDFILTER);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

3.15 TY_BOOL_SGBM_LRC

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_SGBM_LRC, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_SGBM_LRC);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

3.16 TY_INT_SGBM_LRC_DIFF

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_LRC_DIFF, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_LRC_DIFF);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.17 TY_INT_SGBM_MEDFILTER_THRESH

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_MEDFILTER_THRESH, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_MEDFILTER_THRESH);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.18 TY_INT_SGBM_SEMI_PARAM_P1_SCALE

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_SEMI_PARAM_P1_SCALE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_SEMI_PARAM_P1_SCALE);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();
```

```
Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.19 TY_ENUM_DEPTH_QUALITY

```
DevParam param = cl.DevParamFromEnum(TY_DEPTH_QUALITY_HIGH);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_ENUM_DEPTH_QUALITY, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_ENUM_DEPTH_QUALITY);

uint m_read_param = read_param.toEnum();

Console.WriteLine($"current value {m_read_param}");

EnumEntryVector m_read_param2 = read_param.eList();

for (int i = 0; i < m_read_param2.Count(); i++)

{

Console.WriteLine($"{ m_read_param2[i].value}");

}
```

3.20 TY_INT_TOF_CHANNEL

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_CHANNEL, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_CHANNEL);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.21 TY_INT_TOF_MODULATION_THRESHOLD

```
DevParam param = cl.DevParamFromInt(5);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_MODULATION_THRESHOLD, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_TOF_MODULATION_THRESHOLD);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");
```

```
int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.22 TY_INT_TOF_ANTI_SUNLIGHT_INDEX

```
DevParam param = cl.DevParamFromInt(2);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_TOF_ANTI_SUNLIGHT_INDEX, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_TOF_ANTI_SUNLIGHT_INDEX);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.23 TY_INT_MAX_SPECKLE_SIZE

```
DevParam param = cl.DevParamFromInt(200);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_MAX_SPECKLE_SIZE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_MAX_SPECKLE_SIZE);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.24 TY_INT_MAX_SPECKLE_DIFF

```
DevParam param = cl.DevParamFromInt(200);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_MAX_SPECKLE_DIFF, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,
```

```
TY_INT_MAX_SPECKLE_DIFF);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

3.25 TY_BOOL_TOF_ANTI_INTERFERENCE

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_BOOL_TOF_ANTI_INTERFERENCE, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_BOOL_TOF_ANTI_INTERFERENCE);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

3.26 TY_INT_SGBM_TEXTURE_THRESH

```
   DevParam param = cl.DevParamFromInt(1088);

   int error = cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_INT_SGBM_TEXTURE_THRESH,

param);

   DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM,

TY_INT_SGBM_TEXTURE_THRESH);

   int m_read_param = read_param.toInt();

   int min = read_param.mMin();

   int max = read_param.mMax();

   int inc = read_param.mInc();

   Console.WriteLine($"min ：{min},max：{max},inc：{inc}+++++{m_read_param}");
```

4 Features Under the *RGB* Component

4.1 TY_BOOL_AUTO_EXPOSURE

```
DevParam param = cl.DevParamFromBool(false);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_EXPOSURE, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_EXPOSURE);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

4.2 TY_INT_EXPOSURE_TIME

```
DevParam param = cl.DevParamFromInt(500);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_EXPOSURE_TIME, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_EXPOSURE_TIME);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

4.3 TY_BOOL_AUTO_GAIN

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_GAIN, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_GAIN);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

4.4 TY_BOOL_AUTO_AWB

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_AWB, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_BOOL_AUTO_AWB);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

4.5 TY_INT_R_GAIN

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_R_GAIN, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_R_GAIN);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

4.6 TY_INT_G_GAIN

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_G_GAIN, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_G_GAIN);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

4.7 TY_INT_B_GAIN

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_B_GAIN, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_B_GAIN);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

4.8 TY_INT_ANALOG_GAIN

```
DevParam param = cl.DevParamFromInt(1);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_ANALOG_GAIN, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_ANALOG_GAIN);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

4.9 TY_STRUCT_AEC_ROI

```
PercipioAecROI roi = new PercipioAecROI(0, 0, 640, 480);

DevParam param1 = cl.DevParamFromPercipioAecROI(roi);

 cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_STRUCT_AEC_ROI, param1);

 DevParam readParam = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_STRUCT_AEC_ROI);

ArrayVector mReadParam = readParam.toArray();

Console.WriteLine("aec roi: " + string.Join(",", mReadParam));
```

4.10 TY_INT_AE_TARGET_Y

```
DevParam param = cl.DevParamFromInt(200);

cl.DeviceSetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_AE_TARGET_Y, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_RGB_CAM, TY_INT_AE_TARGET_Y);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

5 Features Under the *IR* Component

5.1 TY_INT_GAIN

```
DevParam param = cl.DevParamFromInt(500);

cl.DeviceSetParameter(handle, TY_COMPONENT_IR_CAM_LEFT, TY_INT_GAIN, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_IR_CAM_LEFT, TY_INT_GAIN);

int m_read_param = read_param.toInt();

Console.WriteLine($"current value {m_read_param}");

int min = read_param.mMin();

int max = read_param.mMax();

int inc = read_param.mInc();

Console.WriteLine($"min ：{min},max：{max},inc：{inc}");
```

5.2 TY_BOOL_HDR

```
DevParam param = cl.DevParamFromBool(true);

cl.DeviceSetParameter(handle, TY_COMPONENT_IR_CAM_LEFT,TY_BOOL_HDR, param);

DevParam status = cl.DeviceGetParameter(handle, TY_COMPONENT_IR_CAM_LEFT, TY_BOOL_HDR);

bool m_status = status.toBool();

Console.WriteLine($"current value {m_status}");
```

5.3 TY_BYTEARRAY_HDR_PARAMETER

```
ByteArrayVector array = new ByteArrayVector { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 0, 0, 0, 14,

0, 0, 0 };

DevParam arr = cl.DevParamFromByteArray(array);

cl.DeviceSetParameter(handle, TY_COMPONENT_IR_CAM_LEFT, TY_BYTEARRAY_HDR_PARAMETER,arr);

DevParam hdr_arry = cl.DeviceGetParameter(handle, TY_COMPONENT_IR_CAM_LEFT,

TY_BYTEARRAY_HDR_PARAMETER);

ByteArrayVector hdr_arry_1 = hdr_arry.toByteArray();

for (int i = 0; i < hdr_arry_1.Count(); i++){

   Console.Write($",{hdr_arry_1[i]}");

 }
```

5.4 TY_FLOAT_EXPOSURE_TIME_US

```
DevParam param2 = cl.DevParamFromFloat(1088);

int error2 = cl.DeviceSetParameter(handle, TY_COMPONENT_IR_CAM_LEFT, TY_FLOAT_EXPOSURE_TIME_US,

param2);

DevParam read_param2 = cl.DeviceGetParameter(handle, TY_COMPONENT_IR_CAM_LEFT,

TY_FLOAT_EXPOSURE_TIME_US);

float m_read_param2 = read_param2.toFloat();

Console.WriteLine($"current value {m_read_param2}");

float min2 = read_param2.fMin();

float max2 = read_param2.fMax();

float inc2 = read_param2.fInc();

Console.WriteLine($"min ：{min2},max：{max2},inc：{inc2}");
```

6 Other Common Features

6.1 TY_ENUM_IMAGE_MODE

```
DevParam param = cl.DevParamFromEnum(TY_PIXEL_FORMAT_DEPTH16 | TY_RESOLUTION_MODE_1280x960);

cl.DeviceSetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_ENUM_IMAGE_MODE, param);

DevParam read_param = cl.DeviceGetParameter(handle, TY_COMPONENT_DEPTH_CAM, TY_ENUM_IMAGE_MODE);

uint m_read_param = read_param.toEnum();

Console.WriteLine($"current value {m_read_param}");

EnumEntryVector m_read_param2 = read_param.eList();

for (int i = 0; i < m_read_param2.Count(); i++)

{

        Console.WriteLine($"description {m_read_param2[i].description}, value { m_read_param2[i].value} ");

}
```

6.2 TYSetLogLevel

```
TYSetLogLevel(5);

// Log information output by the SDK on the console. Log level definition:

// 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,
```

6.3 TYSetLogPrefix

```
TYSetLogPrefix(“Percipio Csharp SDK”);

// Check if the log prefix on the console is ”Percipio Csharp SDK”
```

Percipio.XYZ is an industry leading provider of 3D cameras. We provide a broad range of 3D camera products to meet requirements from various applications, such as industrial, automotive, inspection, logistics, medical, education, security and commercial etc. We will continue to develop and optimize our product roadmap to support more 3D vision applications.

Percipio is an independent vendor of 3D machine vision solutions. We provide products and services to system integration customers rather than end users. This marketing strategy allows us to serve multiple sectors and segments, and also means that our success will be based on our customer's success. Together with our customer's industry specific expertise, we can support end users with implementing machine intelligence, which will improve productivity and/or reduce cost.

Affordable 3D Machine Vision

Contact Us

Purchase : info@pcp3d.com
Technical : support@pcp3d.com
Website : www.pcp3d.com
Documentation : doc.percipio.xyz/cam/latest/en/

Statement

* Data mentioned in this document is subject to change without notice.
* The data mentioned in this document may vary due to environmental factors and other reasons. Percipio does not assume any consequences arising therefrom.

YouTube