Communication Interface Specification

ZA1 Type 3D Camera
LA9547A

Ver. 1.0

Visionary Business Center
MEMS Business Promotion Department
The NIPPON SIGNAL CO., LTD.

The Nippon Signal Co., Ltd
LX9657A-001

Modification History The modified section is marked with "☆" to indicate that the section is different from the previous version.

Ver.	Reason for the modification	Modified page, Modification item	Person in charge	Date modified
1	New	—	Yamada	2014.5.26

Table of Contents

Approved by

Checked by
Created by


1. Overview

   PixelSoleil ZA1 (refered to as 3D camera henceforth) is a distance measuring sensor that emits near infrared LED modulated radiation into the space. It measures the distance to the target based on the light return flight time (flight time measuring method = TOF: Time Of Flight), and outputs two-dimensional point image data (distance, brightness, modulated light intensity).
   By connecting the 3D camera and the processing unit (referred to as “host” henceforth) through Ethernet, 3D camera control and image data transfer are performed. This specification describes the communication interface and the communication format between the 3D camera and the host.

2. Interface Specification

Item	Specification
Communication standard (Physical/data link layer)	Ethernet 10Base-T / 100Base-TX (Support of Auto Negotiation)
DHCP client function	Available (Factory setting: Disabled)
IP address	(Fixed) Acquired through 192.168.0.80 or DHCP
Port number	50000 (TCP)
MDI/MDI-X	Support of Auto MDI/MDI-X

   When the DHCP client function is enabled, the 3D camera requests for the IP address from the DHCP server on the network after the power supply is turned on. If no response is received from the DHCP server, request timeout occurs and the IP address that is set in the 3D camera is utilized. Network setting can be changed by the dedicated software or a function. To enable the modified network setting, the 3D camera must be restarted. The camera can be restarted by turning on the power supply again.

   Only one host can be connected to the 3D camera each time.
   The camera is connected via TCP to the port number that is opened by the 3D camera.

3. Software Developer Kit

3.1 Overview

This software developer kit (referred to as SDK henceforth) is a library that is created to enable programmers to create any applications without having to be aware of the internal control of the camera.

The kit is provided in the form of the dynamic link library (DLL).
The use of the kit under Windows® 7 is assumed.
Operation under VC++ is verified. The use under Microsoft® Visual C++ .Net 2008 or later is recommended unless there is a special reason in the development environment or the platform.

Supplied files

File required at application execution

TCS_Dll.dll	Library file Copy the file into the same folder as that of the program that uses DLL, the system folder of “Windows”, or the folder under the same path.

Files required at application development (Used at development with VC++)

TCS_Dll.lib	Library reference file (used at link under VC++) Normally, copy the files into the same folder as that of the source file that is used for program development.
TCS_DllAPI.h	Library function & constant definition file (used by including with VC++) Normally, copy the files into the same folder as that of the source file that is used for program development.

SDK Manual

TCS_SDK.chml	Help file This file. Use this file for development since it contains function descriptions.

Data format

The Word data has 16 bits per pixel.
The data in one frame is arranged as follows, provided that pixel (x,y) position data is expressed as D(x,y).

D(0,0)	D(1,0)	D(2,0)	….	D(126,0)	D(127,0)
D(0,1)	D(1,1)	D(2,1)	….	:	:
D(0,2)	:	:	….	:	:
:	:	:	….	:	:
:	:	:	….	:	:
:	:	:	….	:	:
D(0,127)	D(1,127)	D(2,127)	….	D(126,127)	D(127,127)

3.2 Function List

The following functions are available for SDK.

Category	Function name	Function
Camera initialization/closing	TCS_Initial	Camera initialization processing
	TCS_Close	Camera closing processing
Capture start/stop	TCS_CameraStart	Starts image creation
	TCS_CameraStop	Stops image creation
Reading capture image	TCS_ReadDepthImg	Reads depth image data
	TCS_ReadBrightImg	Reads brightness image data
	TCS_ReadIntensityImg	Reads modulated light intensity image data
Power setting	TCS_SetDepthImgLED_Power	Sets the power of LED at depth image creation
	TCS_SetBrightImgLED_Power	Sets the power of LED at brightness image creation
	TCS_SetBkGndRemove_Gain	Sets removal of background light
Power acquisition	TCS_GetDepthImgLED_Power	Acquires the power of LED at depth image creation
	TCS_GetBrightImgLED_Power	Acquires the power of LED at brightness image creation
	TCS_GetBkGndRemove_Gain	Acquires removal of background light
Distance data setting	TCS_SetIntensityCutOff	Selects the threshold value of depth image clip processing by intensity
	TCS_SetSmrtAccNum	Sets the accumulation count of the Smart Accumulate function
	TCS_SetSmrtAccThres	Selects a threshold value of the Smart Accumulate function
Temperature acquisition	TCS_GetTemperatureLED	Acquires LED temperature
	TCS_GetTemperatureCAM	Acquires camera temperature
Depth image median filter	TCS_SetDepthImgMedianFilter	Sets the use of the depth image median filer
Time stamp	TCS_SetTimeStamp	Sets a time stamp inside of the camera
	TCS_GetTimeStamp	Acquires the time stamp inside of the camera
Indicator setting	TCS_SetIndicator	Sets an indicator
	TCS_GetIndicator	Acquires indicator information
Distance measurement LED modulated frequency setting	TCS_ClockSelect	Changes the frequency for interference prevention

Power control temperature setting at high temperature	TCS_SetHighTempLimit	Sets the temperature at which automatic power control is performed
Conversion from distance to space information	TCS_ConvDtoXYZ	Converts a distance (D) to space information (X,Y,Z)

3.3 Function Description

TCS_Initial

Function name	BOOL TCS_Initial()
Outline	Initializes the hardware of the camera.
Return value	BOOL type Successful TRUE or 1 Failed FALSE or 0
Argument	void None
Supplementary description	Call once at the start of an application.

TCS_Close

Function name	void TCS_Close()
Outline	Performs camera closing processing
Return value	void None
Argument	void None
Supplementary description	Call once at termination of the application.

TCS_CameraStart

Function name	BOOL TCS_CameraStart()
Outline	Starts image generation. Three images are generated concurrently; they are depth image, brightness image, and modulated radiation intensity image. Distance measurement infrared LED of the camera is set to ON.
Return value	BOOL type Successful TRUE or 1 Failed FALSE or 0
Argument	void None
Supplementary description	Generates an image at every 33.3 msec until the image generation stop function TCS_CameraStop() is called. The image is always replaced with the latest one inside of the camera.

TCS_CameraStop

Function name	BOOL TCS_CameraStop()
Outline	Stops image generation. The distance measurement infrared LED of the camera is set to OFF.
Return value	BOOL type Successful TRUE or 1 Failed FALSE or 0
Argument	void None
Supplementary description	The image that is acquired after termination of image generation is not updated from the state immediately before the TCS_CameraStop() function is called.

TCS_ReadDepthImg

Function name	int TCS_ReadDepthImg(WORD *pBuf)	
Outline	Transmits depth image data of one image to the host.	
Return value	int type: The number of data items that were read. Successful Normal at 128 × 128 Failed -1 : Read failure/timeout error 0 : Requested command transmission error	
Argument	WORD *pBuf	Prepare a memory area of 128 × 128 pixels (128 × 128 × 2 bytes) in Word mode and assign a pointer to the area in the argument.
Supplementary description	Always the latest depth image is transmitted. Since the camera updates the image at 30 fps, depth images can be read at the minimum interval of 33.3 msec. Each value that is stored in the argument array pBuf is between 0 and 16383 in 14 bits, which is equivalent to a value between 0 and 15 m. The conversion expressions to the distance will be as follows, where the value that is read from the camera is Val (x,y) and the distance value in mm units is Dist (x,y). When the modulation frequency of infrared LED is 10.0 MHz: Dist(x,y) = 150000 × Val(x,y) ÷ 10.0 ÷ 16384 When the modulation frequency of infrared LED is 10.1 MHz: Dist(x,y) = 150000 × Val(x,y) ÷ 10.1 ÷ 16384	

TCS_ReadDepthBinningImg

Function name	int TCS_ReadDepthBinningImg(WORD *pBuf, INT iLevel)	
Outline	Transmits binned depth image of one image to the host.	
Return value	int type: Number of data items that were read Successful Normal at 128 × 128 Failed -1 : Read failure/timeout error 0 : Requested command transmission error	
Argument	WORD *pBuf	Prepare a memory area of 128 × 128 pixels (128 × 128 × 2 bytes) in Word mode and assign a pointer to the area in the argument.
	INT iLevel	Number of binned pixels. Assign one of 1, 2, and 4.
Supplementary description	Transmits the binned depth images. Each value that is stored in the argument array pBuf is between 0 and 16383 in 14 bits, which is equivalent to a value between 0 and 15 m. The conversion expressions to the distance will be as follows, where the value that is read from the camera is Val (x,y) and the distance value in mm units is Dist (x,y). When the modulation frequency of infrared LED is 10.0 MHz: Dist(x,y) = 150000 × Val(x,y) ÷ 10.0 ÷ 16384 When the modulation frequency of infrared LED is 10.1 MHz: Dist(x,y) = 150000 × Val(x,y) ÷ 10.1 ÷ 16384 When argument iLevel is 1, binning is invalid and the same data as TCS_ReadDepthImg is output. When the argument is 2, binning is performed under 2 × 2 pixels and when the argument is 4, binning is performed under 4 × 4 pixels. All the numbers of pixels that are output remain 128 × 128 and the same value is output as the binned pixels. For the details of binning, refer to **12. Binning Function.**	

TCS_ReadBrightImg

Function name	BOOL TCS_ReadDepthImg(WORD *pBuf)	
Outline	Transmits brightness image data of one image to the host.	
Return value	int type: Number of data items that were read Successful Normal at 128 × 128 Failed -1 : Read failure/timeout error 0 : Requested command transmission error	
Argument	WORD *pBuf	Prepare a memory area of 128 × 128 pixels (128 × 128 × 2 bytes) in Word mode and assign a pointer to the area in the argument.
Supplementary description	Always the latest brightness image is transmitted. Since the camera updates images at 30fps, brightness images can be read at the minimum interval of 33.3 msec.	

TCS_ReadIntensityImg

Function name	BOOL TCS_ReadIntensityImg(WORD *pBuf)	
Outline	Transmits modulated light intensity image data of one image to the host.	
Return value	int type: Number of data items that were read Successful Normal at 128 × 128 Failed -1 : Read failure/timeout error 0 : Requested command transmission error	
Argument	WORD *pBuf	Prepare a memory area of 128 × 128 pixels (128 × 128 × 2 bytes) in Word mode and assign a pointer to the area in the argument.
Supplementary description	A modulated light intensity image refers to the modulated power that returns to the camera at distance measurement. The greater the modulated light intensity is, the less the measurement variations become and the lower the modulated light intensity, the greater the measurement variations become. Therefore, the reliability of the measured distance can be checked from the information of the modulated light intensity. Always the latest modulated light intensity image is read. Since the camera updates images at 30 fps, modulated light intensity images can be read at the minimum interval of 33.3 msec.	

TCS_SetDepthImgLED_Power

Function name	int TCS_SetDepthImgLED_Power(BOOL bAuto, WORD nManualValue)	
Outline	Sets the power of infrared LED at depth image generation.	
Return value	int type: 1 : Successful 0 : Failed -1 : Parameter range error of the 2nd argument	
Argument	BOOL bAuto	TRUE: Auto control mode is set. The value of the 2nd argument is ignored. FALSE: Set to value of the 2nd argument.
	WORD nManualValue	Value that is set within the range from 0 to 100.
Supplementary description	This function automatically adjusts to the optimum power by setting the argument bAuto to TRUE and performing auto power control in the internal camera hardware. When setting the power manually, set bAuto to FALSE and enter a value in nManualValue within the range from 0 to 100. The value indicates the percentage of the power, where 100 indicates 100% of LED power and 0 indicates 0% of LED power.	

TCS_SetBrightImgLED_Power

Function name	Int TCS_SetBrightImgLED_Power(BOOL bAuto, WORD nManualValue)	
Outline	Sets the power of the infrared LED at brightness image generation.	
Return value	int type: 1 : Successful 0 : Failed -1 : Parameter range error of the 2nd argument	
Argument	BOOL bAuto	TRUE: Auto control mode is set. The value of the 2nd argument is ignored. FALSE: Set to value of the 2nd argument.
	WORD nManualValue	Value that is set within the range from 0 to 100.
Supplementary description	This function automatically adjusts the power to the optimum power by setting the argument bAuto to TRUE and performing auto power control in the internal camera hardware. To set the power manually, set bAuto to FALSE and enter a value in nManualValue within the range from 0 to 100. The value indicates the percentage of the power, where 100 indicates 100% LED power and 0 indicates 0% LED power.	

TCS_SetBkGndRemove_Gain

Function name	int TCS_SetBkGndRemove_Gain(BOOL bAuto, WORD nManualValue)	
Outline	Set the background light removal ratio.	
Return value	int type: 1 : Successful 0 : Failed -1 : Parameter range error of the 2nd argument	
Argument	BOOL bAuto	TRUE: Auto control mode is set. The value of the 2nd argument is ignored. FALSE: Set to value of the 2nd argument.
	WORD nManualValue	Value that is set within the range from 0 to 9.
Supplementary description	This function sets the background light removal ratio to the optimum ratio by setting the argument bAuto to TRUE and performing auto gain control of the camera hardware internally. To set the background light removal ratio manually, set bAuto to FALSE and enter a value in nManualValue within the range from 0 to 9. For the details of removal of background light, refer to **5. Removing Ambient Light.**	

TCS_GetDepthImgLED_Power

Function name	int TCS_GetDepthImgLED_Power(WORD *nValue)	
Outline	Acquires the power of infrared LED at depth image generation.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *nValue	Assign a pointer to the WORD type variable. The power value (0-100) is stored in this variable.
Supplementary description	Use this function to check the power of the LED that is currently output at the setting of auto power control.	

TCS_GetBrightImgLED_Power

Function name	int TCS_GetBrightImgLED_Power(WORD *nValue)	
Outline	Acquires the power of infrared LED at brightness image generation.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *nValue	Assign a pointer to the Word type variable. The power value (0-100) is stored in this variable.
Supplementary description	Use this function to check the power of the LED that is currently output at the setting of auto power control.	

TCS_GetBkGndRemove_Gain

Function name	int TCS_GetBkGndRemove_Gain(WORD *nValue)	
Outline	Acquires the background light removal ratio.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *nValue	Assign a pointer to the Word type variable. The power value (0-9) is stored in this variable.
Supplementary description	Use this function to check the current background light removal ratio at the setting of auto gain control. For the details of removal of background light, refer to **5. Removing ambient light.**	

TCS_SetInsensityCutOff

Function name	int TCS_SetIntensityCutOff(WORD nIntensity)	
Outline	Select the threshold value of the depth image clip processing by modulated light intensity.	
Return value	int type 1 : Successful 0 : Failed -1 : Argument parameter range error	
Argument	WORD nIntensity	Assign a value between 0 and 15.
Supplementary description	The modulated light intensity threshold value is assigned for clipping the distance value of the pixel of low modulated light intensity to 15 m. Clip processing is performed by the camera hardware internally. For the details of the selectable threshold values, refer to **6. Intensity Trimming Function.**	

TCS_SetSmrtAccNum

Function name	int TCS_SetSmrtAccNum(WORD nAcumNo)	
Outline	Set the accumulation count of the Smart Accumulate function.	
Return value	int type 1 : Successful 0 : Failed -1 : Argument parameter range error	
Argument	WORD nAcumNo	Assign a value between 0 and 15.
Supplementary description	For the details of the Smart Accumulate function, refer to **7. SmartAccumlation Tecnology.**	

TCS_SetSmrtAccThres

Function name	int TCS_SetSmrtAccThres(WORD nAcumThres)	
Outline	Select a threshold value of the modulated light intensity of the Smart Accumulate function.	
Return value	int type 1 : Successful 0 : Failed -1 : Argument parameter range error	
Argument	WORD nAcumThres	Assign a value between 0 and 15.
Supplementary description	For the details of the Smart Accumulate function, refer to **7. SmartAccumlation Technology.**	

TCS_GetTemperatureLED

Function name	int TCS_GetTemperatureLED(DOUBLE *dValue)	
Outline	Acquires the left/right LED ambient temperatures.	
Return value	int type 1 : Successful 0 : Failed	
Argument	DOUBLE *dValue	Prepare a memory area for two temperature data items (2 × 4 byes) in DOUBLE type and assign a pointer to the memory area. Temperatures are stored in the variable.
Supplementary description	This function acquires temperatures of the temperature sensors near the left and right LEDs. Temperature data is stored in the order of left LED and right LED. The unit of temperatures is (°C). For the details of temperatures, refer to **8. Temperature Monitor Function.**	

TCS_GetTemperatureCAM

Function name	int TCS_GetTemperatureCAM(DOUBLE *dValue)	
Outline	Acquires the temperatures of the control board inside of the camera and the sensor board.	
Return value	int type 1 : Successful 0 : Failed	
Argument	DOUBLE *dValue	Prepare a memory area for two temperature data items (2 × 4 byes) in DOUBLE type and assign a pointer to the memory area. Temperatures are stored in the variable.
Supplementary description	This function acquires the temperatures of the temperature sensors of the sensor board and the control board inside of the camera. Temperature data is stored in the order of the control board and the sensor board. The unit of temperatures is (°C). For the details of temperatures, refer to **8. Temperature Monitor Function.**	

TCS_SetDepthImgMedianFilter

Function name	int TCS_SetDpthImgMedianFilter(BOOL bFlag)	
Outline	Applies a median filter of 3x3 to the depth image.	
Return value	int type 1 : Successful 0 : Failed	
Argument	BOOL bFlag	A median filter is used when TRUE is set. A median filter is not used when FALSE is set.
Supplementary description	For noise processing by a median filter, the values of distances of surrounding 3 × 3 pixels of the target pixels are sorted and the median value is set in the distance value of the target pixel. The use of a median filter is set as the default setting at activation. Remove the setting when a median filter is not required.	

TCS_SetTimeStamp

Function name	int TCS_SetTimeStamp(WORD *dValue)	
Outline	Sets the time in the camera in the time stamp.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *dValue	Prepare a Word type memory area for 7 items (7 × 2 bytes) and assign a pointer to the area. The variable time is transmitted to the camera.
Supplementary description	After the command transmission, the transmitted time is set as the current time of the camera and time counting continues. The time is counted in the unit of 10 ms. The time is stored in the order of year (0 for 2000), month, day, hour (24-hour), minute, second, and millisecond.	

TCS_GetTimeStamp

Function name	int TCS_GetTimeStamp(WORD *dValue)	
Outline	Acquires the time of the time stamp in the camera.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *dValue	Prepare a Word type memory area for 7 items (7 × 2 bytes) and assign a pointer to the area. The variable time is transmitted to the camera.
Supplementary description	This function transmits the command that was received in the camera to the host. The time is counted in the unit of 10 ms. The time is stored in the order of year (0 for 2000), month, day, hour (24-hour), minute, second, and millisecond.	

TCS_SetIndicator

Function name	Int TCS_SetIndicator(WORD *dValue)	
Outline	Sets the LED of the indicator of the camera.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *dValue	Prepare a Word type memory area for 6 items (6 × 2 bytes) and assign a pointer to the area. The variable value is transmitted to the camera.
Supplementary description	This function sets the states of the LED of the three indicators (red, yellow, and green) at the bottom front of the camera. The data is stored in the order of red ON time, red OFF time, yellow ON time, yellow OFF time, green ON time, and green OFF time. For the details of the indicators and arguments, refer to **9. Indicator LED.**	

TCS_GetIndicator

Function name	Int TCS_GetIndicator(WORD *dValue)	
Outline	Acquires the camera indicator LED setting information.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD *dValue	Prepare a Word type memory area for 6 items (6 × 2 bytes) and assign a pointer to the area. The variable value is transmitted to the camera.
Supplementary description	This function sets the states of the LED of the three indicators (red, yellow, and green) at the bottom front of the camera. The data is stored in the order of red ON time, red OFF time, yellow ON time, yellow OFF time, green ON time, and green OFF time. For the details of the indicators and arguments, refer to **9. Indicator LED.**	

TCS_LEDClockSelect

Function name	Int TCS_LEDClockSelect(WORD m_nClockType)	
Outline	Changes the light emission frequency.	
Return value	int type 1 : Successful 0 : Failed	
Argument	WORD m_nClockType	When the value is 0, the light emission frequency is 10.0 MHz. When the value is 1, the light emission frequency is 10.1 MHz.
Supplementary description	When the objects of the same direction are photographed by a number of TOF cameras of the same light emission frequency, the infrared lights that are irradiated mutually interfere, fluctuating the distance measurement significantly. By changing the light emission frequency, the interference can be reduced.	

TCS_SetHighTempLimit

Function name	Int TCS_SetHighTempLimit(WORD dValue)	
Outline	Sets a temperature for LED power control when the camera is at a high temperature.	
Return value	int type 1 : Successful 0 : Failed -1 : Argument parameter range error	
Argument	WORD dValue	Temperature to be set between 0 and 127. The unit is °C.
Supplementary description	To prevent overheating due to the heat generated from the inside of the camera (in particular, LED), the automatic LED power control function is incorporated in the camera. The function controls the LED power to prevent the temperature from exceeding the temperature specified by the argument in the camera. When the camera is activated, the temperature is set to 85°C as the default value. For the LED power control at high temperature, refer to **10. LED Power Control at High Temperature.**	

TCS_ConvDtoXYZ

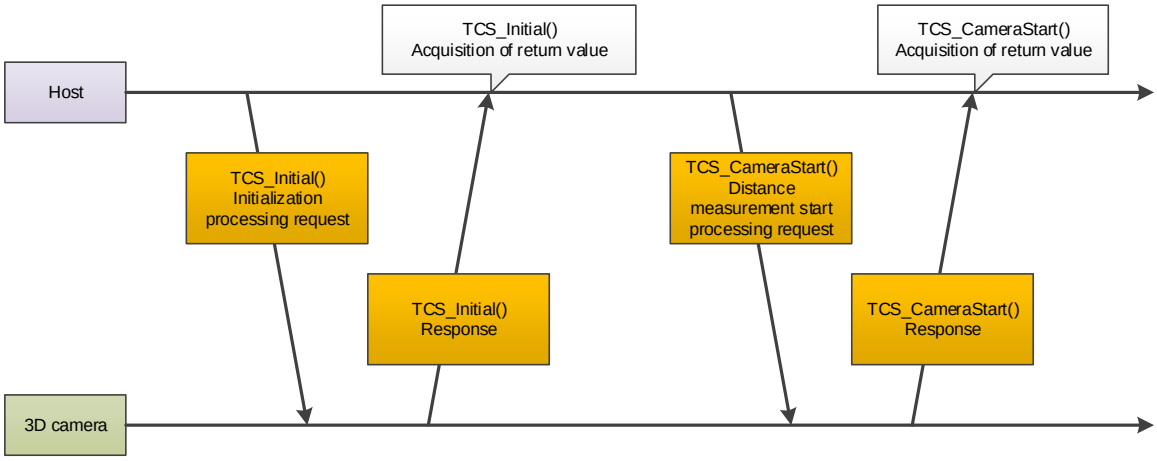
Function name	Int TCS_ConvDtoXYZ(FLOAT *pDImg, FLOAT *pXImg, FLOAT *pYImg, FLOAT *pZImg)	
Outline	Converts a depth image (DImg) to space information (XImg, YImg, ZImg).	
Return value	int type 1 : Successful 0 : Failed	
Argument	FLOAT *pDImg	Assign a pointer to the depth image D image.
	FLOAT *pXImg	Assign a pointer to the X image after conversion to space information.
	FLOAT *pYImg	Assign a pointer to the Y image after conversion to space information.
	FLOAT *pZImg	Assign a pointer to the Z image after conversion to space information.
Supplementary description	The distance information that is transmitted from the camera is the absolute distance from the object that is projected on the target pixel. Therefore, when a plane such as a wall is photographed, a different distance is output depending on the view angle of the receiver lens at the center and around the image. This function perform a conversion of the absolute distance (D) from the camera (pixel) and the relative position (X, Y, Z) on the three-dimensional space based on the camera as the origin point, by adding the view angle information of the lens. As a result, when a plane such as a wall is photographed, the space information of the Z image indicates the same distance in all the pixels. For the details of conversion of the D image, refer to **11. XYZ Conversion of D Image.** For the D image, use the phase distance conversion expression according to the infrared LED modulated frequency that is described in the TCS_GetDepthImg function.	

4. Examples of Communication Control

This section shows the flow from the establishment of the host communication path of the 3D camera to acquisition of image data and termination of processing in the form of communication sequence.

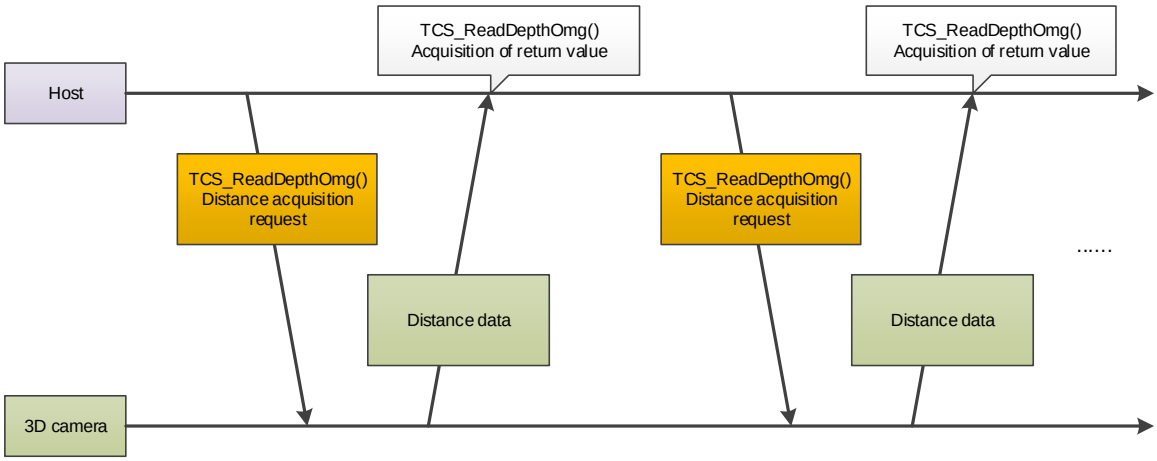
4.1 Communication Establishment

Start measurement by using initialization function “TCS_Initial” and distance acquisition start function “TCS_CameraStart” of the 3D camera.



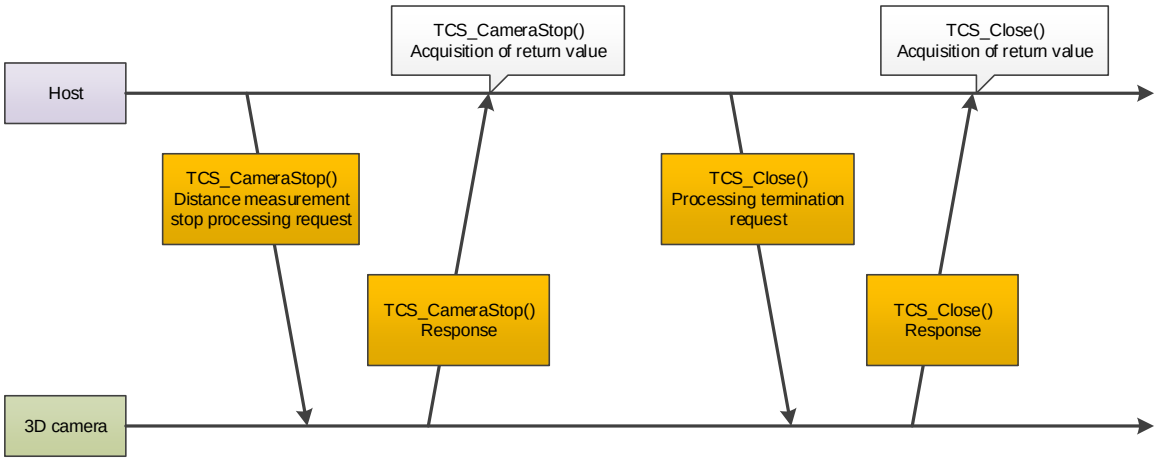
4.2 Acquiring depth image data

Acquire depth image data by using distance acquisition function “TCS_ReadDepthImg”. To acquire data consecutively, call TCS_ReadDepthImg for each frame.



4.3 Processing Termination

Use distance acquisition termination function “TCS_CameraStop” and processing termination function “TCS_Close” to terminate the processing.



5. Removing Ambient Light

The image sensor of the 3D camera is equipped with a powerful built-in ambient light removal circuit in each pixel, removing ambient light at the pixel level.

The higher the ambient light removal ratio is, the more difficult for the image sensor to be saturated, increasing the ambient light resistance performance. However, the charge loss within the pixel increases and as a result, the distance measurement precision deteriorates slightly. Therefore, it is necessary to set the optimum ambient light removal ratio for the photographing environment.

The hardware of the 3D camera is equipped with the auto gain function that automatically sets the ambient light removal ratio based on the image data that is photographed.

Auto gain is set to ON as the default setting when the power is turned on.

The ambient light removal ratio can also be adjusted manually from the host.

By using the TCS_SetBkGndRemove_Gain function, the background light removal ratio setting can be switched between Auto and Manual and the background light removal ratio can be specified.

In manual mode, specify the background light removal ratio setting argument within the range from 0 to 9. See Table 5-1 for the relationship between argument values and the background light removal ratios.

Table 5-1 Relationship between arguments and background light removal ratios

Argument	Background light removal ratio
0	About 2:1
1	About 4:1
2	About 6:1
3	About 8:1
4	About 10:1
5	About 12:1
6	About 14:1
7	About 16:1
8	About 18:1
9	About 20:1

6. Intensity Trimming Function

   Since the distance values of pixels of low modulated light intensity vary significantly and thus the reliability is low, distances different from actual distances may be output.
   The 3D camera is equipped with the function that clips the distances of pixels of low modulated light intensity and the intensity threshold value for clipping can be specified by using the TCS_SetIntensityCutOff function.
   By using the intensity trimming function, depth images of low deviations can be obtained by removing pixels of low reliability.

   Assign a value between 0 and 15 for the argument of the TCS_SetIntensityCutOff function. See Table 6-1 for the relationship between the arguments and the intensity threshold values that are assigned. The default value that is set in the hardware after power ON is 0 (trimming OFF).

Table 6-1 Relationship between arguments and intensity threshold values

Argument	Trimming intensity threshold value HEX(DEC)
0	Trimming OFF
1	0x0008 (8)
2	0x0010 (16)
3	0x0020 (32)
4	0x0030 (48)
5	0x0040 (64)
6	0x0060 (96)
7	0x0080 (128)
8	0x00A0 (160)
9	0x00D0 (208)
10	0x0100 (256)
11	0x0140 (320)
12	0x0180 (384)
13	0x01C0 (448)
14	0x0200 (512)
15	0x0280 (640)

7. SmartAccumulation Technology

   SmartAccumlation Technology is the technology that controls distance variation of the low-intensity pixels. This technology adds pixels (each phase information) tracing back until the accumulated modulated light intensity of the target pixel exceeds the threshold value.
   Specify the threshold value of accumulated modulated light intensity by using the TCS_SetSmrtAccThres function.
   Specify the argument of the TCS_SetSmrtAccThres() function within the range from 0 to 15. See Table 3.2 for the relationship between arguments and threshold values of the accumulated modulated light intensity that are assigned to the camera.
   Since pixels of low modulated light intensity are added by tracing back until the accumulated modulated light intensity reaches the threshold value, time resolution deteriorates. As a result, the moving object in the image leaves a trail. To prevent such an effect, the maximum number of past images that are added can be specified by using the TCS_SetSmrtAccNum() function. The argument indicates the maximum number of past images that are added and past images within the range from 0 to 15 are added until the number of images exceeds the accumulated modulated light intensity threshold value that is specified in TCS_SetSmartAccThres. Addition is not performed for the pixels of the intensity higher than the accumulated modulated light intensity threshold value that is specified in TCS_SetSmartAccThres.

SmartAccumlation Technology enables control of distance variations of low intensity pixels (objects of far distance and objects of low reflection rate) while maintaining the time resolution of comparatively high intensity pixels (objects of close distance and objects of high reflection rate). Even if accumulation addition is performed by SmartAccumlation, the depth image output rate of 30 fps remains unchanged.
The intensity threshold default value that is set in the hardware is 15 (no threshold value) at power ON and the number of picture images that are added is 2.

Table 7-1 Arguments and intensity threshold values of SmartAccumlation

Argument	Accumulated modulated light intensity threshold value HEX (DEC)
0	0x0000 (SmartAcum OFF)
1	0x0020 (32)
2	0x002C (44)
3	0x003E (62)
4	0x0057 (87)
5	0x007A (122)
6	0x00AB (171)
7	0x00F0 (240)
8	0x0150 (336)
9	0x01D6 (470)
10	0x0292 (658)
11	0x0399 (921)
12	0x050A (1290)
13	0x070E (1806)
14	0x09E0 (2528)
15	No threshold value (added up to the maximum number of images)

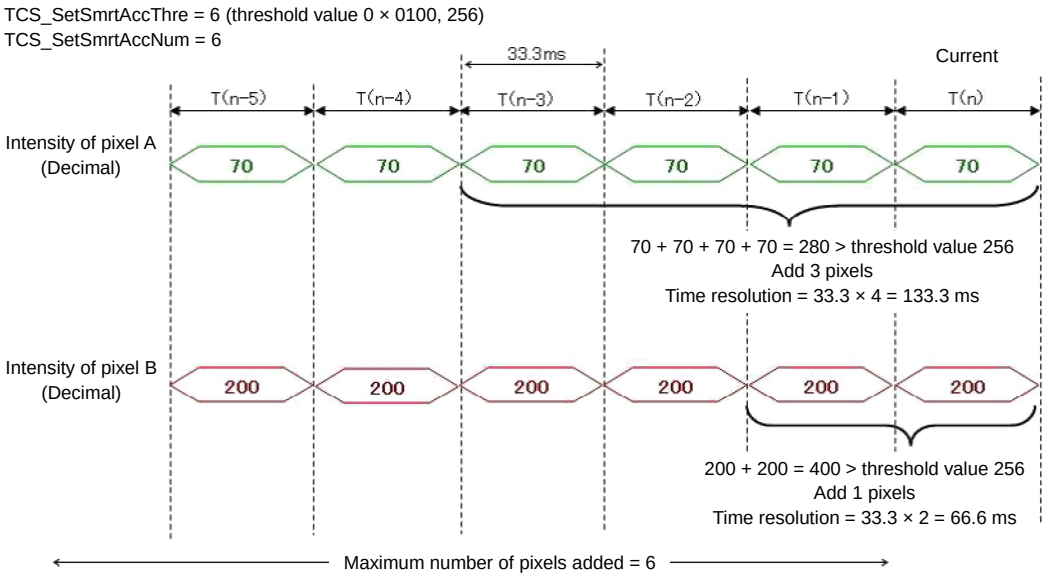


Figure 7-1 SmartAccumlation1

Pixels are added tracing backwards until the intensity exceeds the intensity threshold value (256 in Figure 7-1). Addition processing is performed for each pixel.

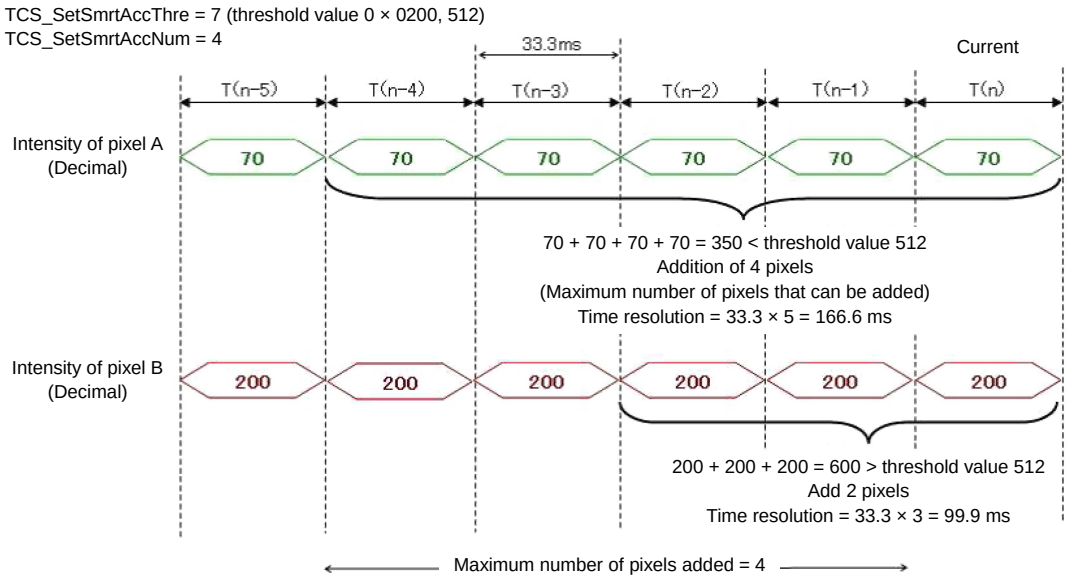


Figure 7-2 SmartAccumlation2

As shown in pixel A in Figure 7-2, pixels will not be added exceeding the maximum number even if the intensity does not exceed the intensity threshold value (512).

8. Temperature Monitor Function

The 3D camera is equipped with a built-in temperature sensor in the sensor board, the control board, and left and right LED boards.
In the 3D camera, the temperature is always monitored and if any of the temperature sensors detects a temperature close to the upper limit temperature (default setting 85 °C) that can be set by the function, the function controls the power of the infrared LED to prevent the temperature from exceeding the upper limit temperature.
It is recommended to use the 3D camera under the ambient temperature range from −10 to 50 °C. When the 3D camera is to be used under a high temperature, it is recommended to monitor the temperature indicated by the sensor on the user side.

The Viewer software issues a warning if any of the temperature sensors indicates a temperature exceeding 80 °C and if the temperature exceeds 85 °C, the software sets the LED to OFF and stops measurement.

9. Indicator LED

The 3D camera is equipped with three LED indicators (green, yellow, and red) and the lights can be turned on and off from the host.
The indicators are set as follows as the default operation immediately after the power of the 3D camera is turned on.

●Indicator 1 Red ⇒ Indicates an error occurrence status

LED operation	Internal status
OFF	No error (normal operation)
ON	Error

●Indicator 2 Yellow ⇒ Indicates that measuring is in progress (infrared LED ON)

LED operation	Internal status
OFF	Stop
ON	Measuring

●Indicator 3 Green ⇒ Indicates the Ethernet communication status

LED operation	Internal status
OFF	No LINK
ON	LINK
Blinking	Communicating

Indicator operation can be set manually by using the TCS_SetIndicator() function. Six arguments are available for the TCS_SetIndicator() function, red ON, red OFF, yellow ON, yellow OFF, green ON, and green OFF. By sending the setting values that are shown in the following table, temperatures can be controlled from the host.

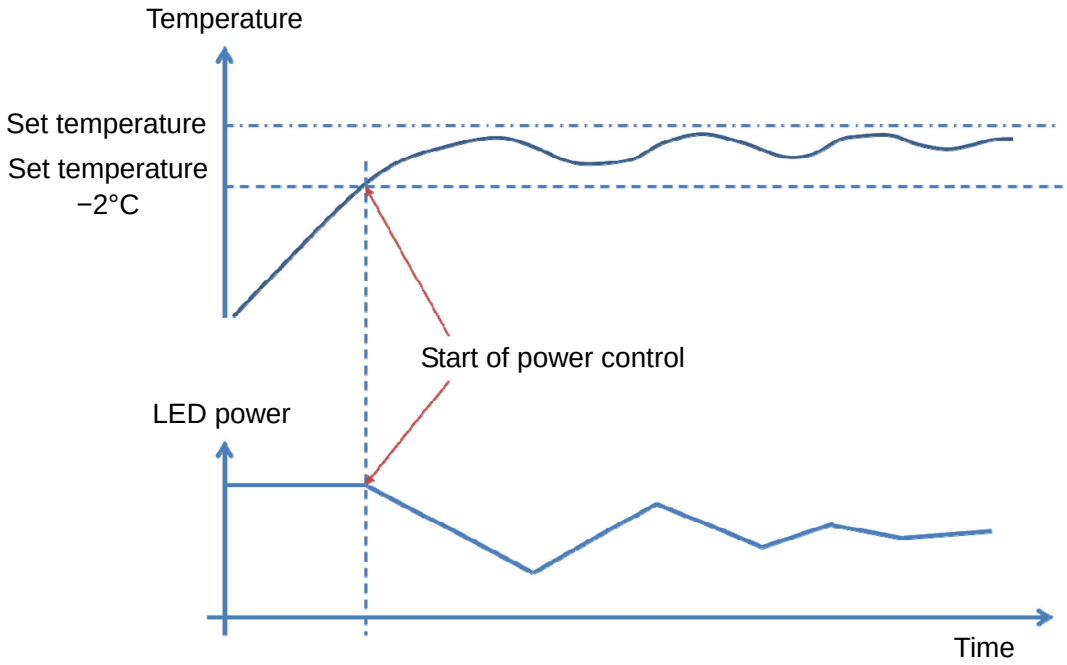
ON value	OFF value	LED operation (common for each color)
0x00	0x00	Default operation of each color indicated above
0x00	0x01	Always OFF
0x01	0x00	Always ON
0x01 to 0x64	0x01 to 0x64	ON/OFF operation with the value that is set × 0.1 second
0x65 to 0xFF	0x65 to 0xFF	Retains the status that is set

10. LED Power Control at High Temperature

The 3D camera generates heat mainly from the infrared LED while measuring a distance. When the ambient temperature of the camera is high (50 °C or higher), the temperature inside of the 3D camera is higher. To prevent heat damage, the camera is equipped with the high temperature LED power control function.

The diagram below shows the actual operation. When the temperature inside of the 3D camera becomes close to the set temperature (85 °C as the default), the function reduces the power of the infrared LED to prevent the temperature from exceeding the set temperature. When the temperature falls below the set temperature and the temperature continuously drops, the function tries to control the power of the infrared LED to the original power.

The set temperature can be changed by using TCS_SetHighTempLimit(). Assign a set temperature (°C) with an integer. Although the temperature can be set within the range from 0 to 127 °C, it is recommended to set at 85 °C or lower.



If the temperature rises sharply, the temperature may exceed the set temperature. It takes a time for power control to stabilize around the set temperature.

If the LED power control functions normally, reducing the power, measured distance variations increase. Use the camera under the ambient temperature or the heat release environment away from the set temperature as much as possible.

11. XYZ Conversion of D Image

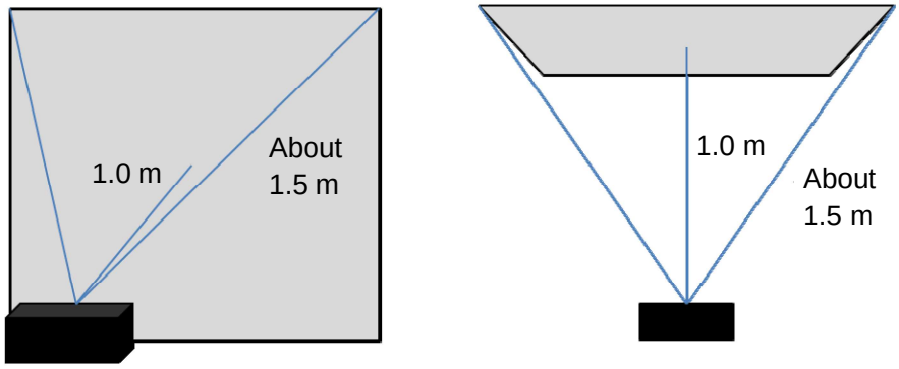
11.1 D image

Distance information that is output from the 3D camera indicates the absolute distance from the object that is projected in the target pixel.

That is, when a distance of a plane such as a wall as shown in the diagram below is measured, the distance value of the pixel near the center of the image is almost the same as the distance up to the wall. However, for the distance value of the pixel around the image, a distance value further than the distance up to the wall is output due to the influence of the view angle of the 3D camera.

For instance, when all the horizontal/vertical angles are 72 degrees and the distance of the wall of 1.0 m is measured, 1.0 m is output at the center of the image, however, the distances that are output from the four corners of the image will be about 1.5 m.

The image that indicates the absolute distance from the target pixel is referred to as D (Distance) image.



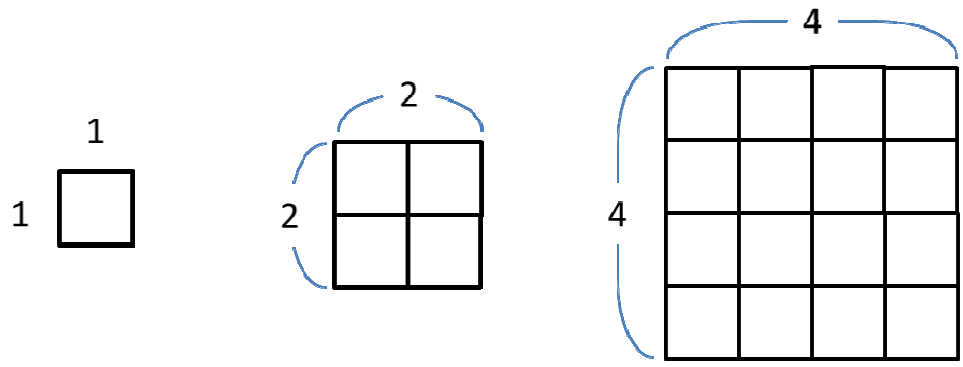
11.2 XYZ image

The XYZ image refers to the result obtained by plotting D image information in the three-dimensional space and calculating the X, Y, and Z coordinate values for each pixel. View angle information is required for converting D image to XYZ image. In the D camera, the view angle of the lens that is mounted is automatically added for conversion.

By using the Z image in the XYZ image, plane information can be obtained by eliminating the distance variations within the same plane, which exist in the D image.

12. Binning Function

The binning function reduces variations by applying averaging among the adjacent pixels.
By using the TCS_ReadDepthBinningImg() function, binned images can be acquired.
A binning pixel can be specified by the iLevel argument. Assign any of 1, 2, and 4. As shown below, binning is performed based on 1 × 1 (no binning), 2 × 2 pixels, or 4 × 4 pixel.



After binning, the number of pixels is reduced by as many as those that were binned. That is, under 1 × 1 (no binning), the image is output under 128 × 128 pixels, under 2 × 2 binning, the image is output under 64 × 64 pixels, and under 4 × 4 binning, the image is output under 32 × 32 pixels.