

# **Application Program Interface Reference Manual**

**for TOF Sensor  
Software Development Kit (SDK)**

**Version 2.2.x**

**Revision 2.2**

**July 12, 2018**

**Hitachi-LG Data Storage, Inc.**

## Contents

<b>1</b>	<b>Introduction.....</b>	<b>1</b>
1.1	Purpose .....	1
1.2	Scope .....	1
1.3	Related Documents.....	1
1.4	Trademarks .....	1
<b>2</b>	<b>Basic Knowledge .....</b>	<b>2</b>
2.1	About TOF .....	2
2.2	Comparision of Distance Sensors.....	4
<b>3</b>	<b>Hardware .....</b>	<b>6</b>
3.1	Hardware Specification .....	6
3.2	Hardware Types .....	7
3.3	TOF ID.....	8
<b>4</b>	<b>Getting Started.....</b>	<b>9</b>
4.1	License .....	9
4.2	System Requirement.....	10
4.2.1	Development Environment .....	10
4.2.2	Operational Environment.....	10
4.3	Compatibility .....	11
4.3.1	Hardware Compatibility .....	11
4.3.2	Software Compatibility (for Windows) .....	11
4.4	Setup Development Environment.....	13
4.4.1	Windows7/8/8.1/10.....	13
4.4.2	Ubuntu14.04 .....	20
4.4.3	Ubuntu 16.04.....	23
4.4.4	Debian 8.0 .....	26
4.4.5	CentOS 7.0 .....	30
4.5	Setup TOF Sensor.....	33
4.5.1	Setup and start .....	33
4.5.2	Setting IP address.....	34
4.5.3	Switch Hardware Type.....	36
4.5.4	Interference Prevention .....	38
4.5.5	Setting Upside Down (HLS-LFOM5) .....	40
4.5.6	Time .....	41
4.5.7	Register to initialization file .....	42
<b>5</b>	<b>Software Configuration.....</b>	<b>43</b>
5.1	System Configuration .....	43
5.2	System Performance .....	44
5.3	Coordinate System.....	46
5.4	Detection Distance.....	47
5.4.1	Distance Mode .....	47
5.5	Convert to Distance .....	48
5.6	Coodinate Rotation .....	49
5.7	Image Size .....	51
5.8	Frame Data .....	52
5.9	Class.....	53
5.10	State Transition Diagram .....	54

5.11 Sequence .....	55
5.11.1 Start Sequence .....	55
5.11.2 Read Frame Sequence .....	56
5.12 Initialization File.....	58
5.13 Features.....	59
5.13.1 Human Detection .....	59
5.13.2 Capture/Emulation .....	61
5.13.3 Background Subtraction .....	63
5.13.4 Noise Reduction .....	65
<b>6 API Reference .....</b>	<b>68</b>
6.1 TofManager Class .....	68
6.1.1 TofManager::Initialize() Method .....	69
6.1.2 TofManager::Open() Method .....	70
6.1.3 TofManager::Close() Method .....	70
6.1.4 TofManager::GetTofList() Method .....	71
6.1.5 TofManager::AddTof() Method .....	72
6.1.6 TofManager::DeleteTof() Method .....	73
6.2 Tof Class .....	74
6.2.1 Tof::Open() Method .....	76
6.2.2 Tof::Open() Method .....	76
6.2.3 Tof::Close() Method .....	77
6.2.4 Tof::Run() Method .....	77
6.2.5 Tof::Run() Method .....	78
6.2.6 Tof::Stop() Method .....	78
6.2.7 Tof::Restart() Method .....	79
6.2.8 Tof::SetStandbyMode() Method .....	79
6.2.9 Tof::GetVersion() Method .....	80
6.2.10 Tof::GetStatus() Method .....	80
6.2.11 Tof::GetFrameStatus() Method .....	81
6.2.12 Tof::ReadFrame() Method .....	82
6.2.13 Tof::ReadFrame() Method .....	83
6.2.14 Tof::ReadFrame() Method .....	84
6.2.15 Tof::ReadFrame() Method .....	85
6.2.16 Tof::ReadFrame() Method .....	86
6.2.17 Tof::ReadFrame() Method .....	87
6.2.18 Tof::SetCameraMode() Method .....	88
6.2.19 Tof::GetCameraMode() Method .....	88
6.2.20 Tof::SetCameraPixel() Method .....	89
6.2.21 Tof::GetCameraPixel() Method .....	89
6.2.22 Tof::SetIrGain() Method .....	90
6.2.23 Tof::GetIrGain() Method .....	90
6.2.24 Tof::SetDistanceMode() Method .....	91
6.2.25 Tof::GetDistanceMode() Method .....	91
6.2.26 Tof::SetFrameRate() Method .....	92
6.2.27 Tof::GetFrameRate() Method .....	92
6.2.28 Tof::SetLowSignalCutoff() Method .....	93
6.2.29 Tof::GetLowSignalCutoff() Method .....	93
6.2.30 Tof::SetFarSignalCutoff() Method .....	94

6.2.31 Tof::GetFarSignalCutoff() Method .....	94
6.2.32 Tof::SetEdgeSignalCutoff() Method .....	95
6.2.33 Tof::GetEdgeSignalCutoff() Method .....	95
6.2.34 Tof::ResetBackground() Method .....	96
6.2.35 Tof::SetBackgroundInterval() Method .....	97
6.2.36 Tof::GetBackgroundInterval() Method .....	97
6.2.37 Tof::SetBackgroundQuantity() Method .....	98
6.2.38 Tof::GetBackgroundQuantity() Method .....	98
6.2.39 Tof::CreateCaptureFile() Method .....	99
6.2.40 Tof::Capture() Method .....	100
6.2.41 Tof::GetCaptureStatus() Method .....	100
6.2.42 Tof::PauseEmulationTof() Method .....	101
6.2.43 Tof::SetAttribute() Method .....	102
6.2.44 Tof::GetAttribute() Method .....	103
6.3 FrameData Class .....	104
6.4 FrameMatrix Class .....	105
6.5 FrameDepth Class .....	106
6.5.1 FrameDepth::CreateColorTable() Method .....	107
6.5.2 FrameDepth::Save() Method .....	107
6.5.3 FrameDepth::CalculateLength() Method .....	108
6.6 FrameIr Class .....	109
6.6.1 FrameIr::Save() Method .....	110
6.7 Frame3d Class .....	111
6.7.1 Frame3d::Convert() Method .....	112
6.7.2 Frame3d::Rotate() Method .....	112
6.7.3 Frame3d::RotateZYZ() Method .....	113
6.8 FrameHumans Class .....	114
6.9 Structure .....	115
6.9.1 TofInfo Structure .....	115
6.9.2 TimeStamp Structure .....	115
6.9.3 LensParam Structure .....	116
6.9.4 CaptureInfo Structure .....	116
6.9.5 TofPoint Structure .....	116
6.9.6 Human Structure .....	116
6.10 Enumerated Type .....	117
6.10.1 RunMode Type .....	117
6.10.2 CameraMode Type .....	117
6.10.3 CameraPixel Type .....	117
6.10.4 DistanceMode Type .....	118
6.10.5 FrameRate Type .....	118
6.10.6 OutputData Type .....	118
6.10.7 BgInterval Type .....	119
6.10.8 BgQuantityType .....	119
6.10.9 HumanStatusType .....	120
6.10.10 CaptureStatus Type .....	120
6.10.11 EdgeSignalCutoff Type .....	120
6.10.12 Result Type .....	121
<b>7 Support Tools .....</b>	<b>122</b>

7.1 Reference Code.....	122
7.1.1 Tof2dViewer.cpp.....	122
7.1.2 Tof3dViewer_cv.cpp .....	123
7.1.3 Tof3dViewer_pcl.cpp .....	124
7.1.4 TofIrViewer.cpp.....	125
7.1.5 HumanCounter.cpp.....	126
7.1.6 Tofv2Viewer.cpp.....	130
7.2 Firmware Updater.....	131
<b>8 Terminologies .....</b>	<b>132</b>

## Revision History

Revision	Date	Description
Rev.1.0	Mar 29, 2016	Initial Release
Rev.1.1	April 22, 2016	<ul style="list-style-type: none"> <li>• Support SDK v1.1.0</li> <li>• Add #4.2 System Requirement</li> <li>• Add #4.4 Setup Development Environment</li> <li>• Add #4.5 Setup TOF Sensor</li> <li>• #5.2 Correct FPS (32 -&gt; 30)</li> <li>• #5.5 Correct calculation of convert depth data to actual length</li> <li>• #5.9 Change Class diagram</li> <li>• #5.12 Auto setting for empty items in initialization file</li> <li>• #6.2.1 Cancel Standby in Tof::Open()</li> <li>• #6.2.9 Change argument of Tof::GetVersion()</li> <li>• #6.2.16 Add Tof::ReadFrame(Frame3d*)</li> <li>• #6.4, #6.5 Add change endian.</li> <li>• #6.5.2, #6.6.1 Enabling Save() methods</li> <li>• #6.5.3 Add FrameDepth::CalculateLength()</li> <li>• Delete FrameDepth::ConvertDepthTo3D() method</li> <li>• #6.7 Add Frame3d class</li> <li>• #6.10.12 Add error code in Result type</li> <li>• #7.1 Add sample code</li> </ul>
Rev.1.2	June 3, 2016	<ul style="list-style-type: none"> <li>• Support SDK v1.2.0</li> <li>• #4.1 Change license description</li> <li>• #4.4.1.2 Add installation of Visual Studio 2012/2013 Visual C++redistribution package</li> <li>• #4.5.6,#5.12 Add description if 0 is set to RTP port number</li> <li>• #5.10 Delete Standby from State Transition Diagram</li> <li>• #6.2.7 Add usage of Tof::Restart()</li> <li>• #6.5.2,#6.6.1 Officially support save as JPEG/PNG</li> <li>• #6.6 Change data value of FrameIr</li> <li>• #6.2.24,#6.2.25,#6.10.4 Add changing distance mode</li> <li>• #6.2.26,#6.2.27 Add changing frame rate</li> <li>• #0 Add sample code of TofIrViewer.cpp</li> <li>• #7.1.5 Add sample app. of HumanDetect.exe</li> </ul>
Rev.1.3	July 1, 2016	<ul style="list-style-type: none"> <li>• Support SDK v1.3.0</li> <li>• #4.4.1.1 Add tofd.lib and HumanCounter to deliverables</li> <li>• #5.6 Add coordinate rotation</li> <li>• #5.13.1 Add Human Detection feature</li> <li>• #5.13.2 Add Capture/Emulation feature</li> </ul>

		<ul style="list-style-type: none"> <li>• #6.2.2 Add Open() for Emulated ToF</li> <li>• #6.2.5 Add Run() with Run mode</li> <li>• #6.2.17 Add ReadFrame() for human detection</li> <li>• #6.2.28/6.2.29 Add Set/GetLowSignalCutoff()</li> <li>• #6.2.39/6.2.40/6.2.41 Add methods for capture</li> <li>• #6.2.43/6.2.44 Add Set/GetAttribute()</li> <li>• #6.7.2 Add Rotate() to Frame3d class</li> <li>• #6.8 Add FrameHumans class</li> <li>• #6.9 Add descriptions of structures</li> <li>• #6.9.4 Add CaptureInfo structure</li> <li>• #6.9.6 Add Human structrue</li> <li>• #6.10.1 Add RunMode type</li> <li>• #6.10.9 Add HumanStatus type</li> <li>• #6.10.10 Add CaptureStatus type</li> <li>• #7.1.5 Add HumanCounter.cpp</li> </ul>
Rev.2.0	September 1, 2016	<ul style="list-style-type: none"> <li>• Support SDK v2.0.0</li> <li>• #3.2 Add Hardware Type</li> <li>• #4.3.1,#4.3.2 Add Hardware/Software compatibility</li> <li>• #5.4 Add Detection Distance</li> <li>• #5.7 Add Image Size</li> <li>• #5.8 Add Frame Data</li> <li>• #5.13.2 Add Frame Play and Pause in emulation mode</li> <li>• #5.13.4.1 Add Low Signal Cutoff</li> <li>• #5.13.3 Add Background Subtraction</li> <li>• #6.2.13,#6.2.14 Add ReadFrame() for 2 frames</li> <li>• Add Set/GetDistanceMin()</li> <li>• #6.2.34,#6.2.35,#6.2.36,#6.2.37,#6.2.38 Add Background Subtraction</li> <li>• #6.2.42 Add PauseEmulationToF()</li> <li>• #6.10.1 Add frame by frame play for emulation mode</li> <li>• #6.10.2 Add 2 frame simultaneous mode</li> <li>• #6.10.7, #6.10.8 Add modes for Background Subtraction</li> <li>• #7.1.6 Add Tofv2Viewer.cpp</li> </ul>
Rev.2.1	August 8, 2017	<ul style="list-style-type: none"> <li>• Support SDK v2.1.0</li> <li>• #4.3.1 Add TOFv2 PicByte</li> <li>• #4.3.2 Only for Windows SDK</li> <li>• #4.3.2 Add Visual Studio 2015</li> <li>• #4.3.2 Add OpenCV condition</li> <li>• #4.4.1 Folder structure of v2.1.0</li> <li>• #4.4.1 Add build method of sample programs</li> <li>• #4.4.2~#4.4.5 Add Linux SDK</li> <li>• #5.6,#6.7.3 Add Frame3d::RotateZYX()</li> <li>• #5.13.4 Add Noise Reduction</li> <li>• #5.13.4.2,#6.2.32 /6.2.33 Add Far Signal Cutoff</li> <li>• #5.13.4.3,#6.2.34 /6.2.35 Add Edge Signal Cutoff</li> <li>• Delete DistanceMin()</li> <li>• #7.1.1~7.1.6 Move build method to #4.4</li> <li>• #7.1.3 Change version and usage of PCL</li> <li>• #7.1.5 Add hand detection</li> </ul>
Rev.2.2	June 20, 2018	<ul style="list-style-type: none"> <li>• Support SDK v2.2.0</li> <li>• #3, #4.3.1, #4.5.1 Support HLS-LFOM3 and HLS-LFOM5 hardware type</li> <li>• #4.4 Change package structure</li> <li>• #4.4.1 Add tof_utility tool</li> <li>• #4.5.2 Add HLS-LFOM5 Web GUI for IP address setting</li> </ul>

		<ul style="list-style-type: none"> <li>• #4.5.3 Add hardware type change</li> <li>• #4.5.4 Add interference prevention setting</li> <li>• #4.5.5 Add upside down setting</li> <li>• #6.1 Add bConsoleFlag in TofManager class</li> <li>• #6.2 Add bConsoleFlag in Tof class</li> <li>• #6.10.5 Add fr10fps in FrameRate type</li> <li>• #7.1 Change usage for reference codes</li> </ul>
--	--	--

---

**NOTE:**

*Small changes to improve the accuracy and quality of the information in this document, such as rectifying typographical errors and omissions, will be made without notice.*

---

# 1 Introduction

## 1.1 Purpose

This document is the API reference manual for developers to develop applications which use the software development kit (SDK) of Hitachi-LG Data Storage's TOF sensor.

## 1.2 Scope

This document describes software development kit (SDK) that runs on PC/Server to control TOF sensor. TOF sensor hardware, TOF sensor firmware, and applications which use the software development kit are not described.

## 1.3 Related Documents

Table 1-1 shows related document.

Table 1-1 Related Documents	
Document Name	Remark
TOF Delivery Specification	

## 1.4 Trademarks

- Linux is a registered trademark of Linus Torvalds.
- Other trademarks, registered trademarks, trade names, service marks, and company names may be used in this document. They are properties of their respective owners.

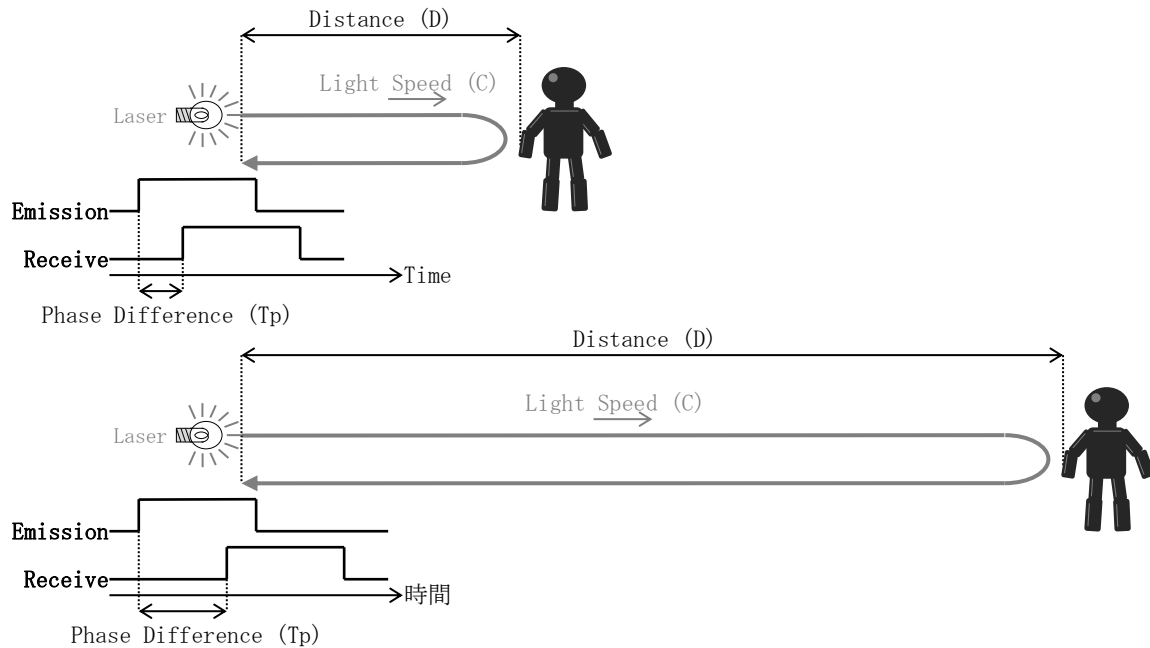


## 2 Basic Knowledge

### 2.1 About TOF

TOF stands for "Time of Flight" that is a technology to measure a distance between objects measuring the time that a sound wave, a ray, a micro wave or other wave travels to the object and returns. It has been widely used for sonar, survey, motion capture, and so on.

Figure 1 shows the principle of TOF using laser beam. When pulsed laser is emitted, there is phase difference ( $T_p$ ) between emitted pulse and returned pulse after reflecting at an object (including a human). The farther distance between the object, the larger phase difference ( $T_p$ ).



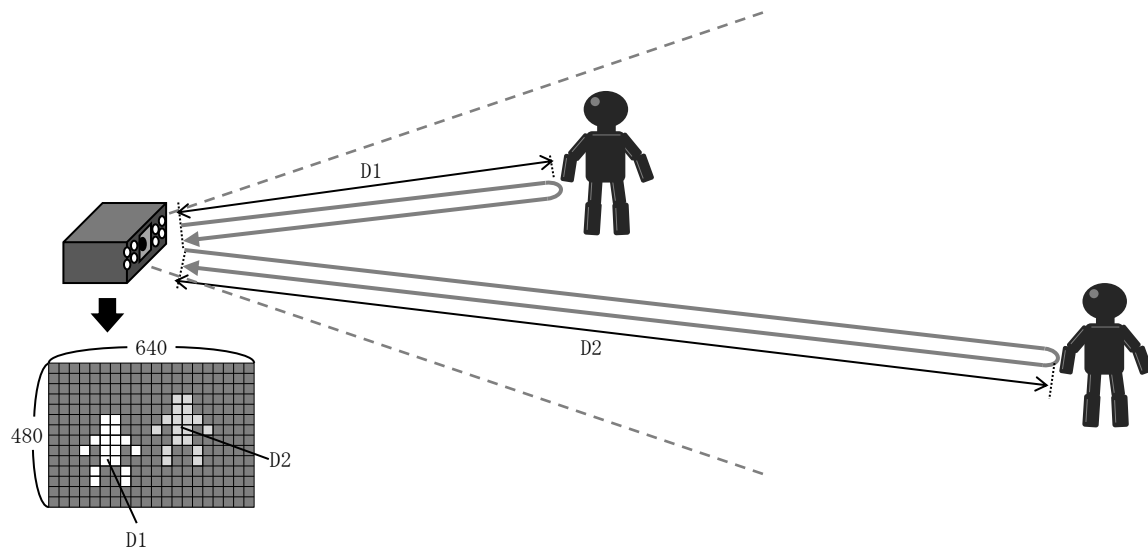
**Figure 1 Principle of TOF**

Since the light speed (C) is **299,792,458m/s (about 300 million m/s)**, if C is light speed,  $T_p$  is phase difference, D of distance between the object is calculated by the following formula.

$$D = \frac{C \cdot T_p}{2}$$

For example, if phase difference ( $T_p$ ) is 10ns, the distance (D) between the object is 1.5m because light moves 3m during 10ns. Since light speed is quite stable, the laser type TOF can measure distance very precisely during very short period, however it requires very high measurement technology in nanosecond order.

The TOF sensor of Hitachi-LG Data Storage irradiates laser light radially, and light receiving elements to measure phase difference are aligned 640 horizontal and 480 vertical so that multiple distances between multiple objects can be measured simultaneously in a space. (Refer to Figure 2)



**Figure 2 TOF Camera**

Measuring depth distance of each pixel of VGA picture (640 x 480) allows to convert captured pictures into 3D. Furthermore, measuring 30 times in a second (30fps) allows to track motions of objects in 3D.

This TOF sensor can perform high efficient and precise sensing using the laser control technology which our company has cultivated in optical disc drive development for long period.

## 2.2 Comparison of Distance Sensors

There are several types of distance sensor shown as Figure 3. There are 2 type of TOF that is for space perception, one is "scan method" that measures moving a thin laser beam, another one is "surface irradiation" that spreads laser beam.

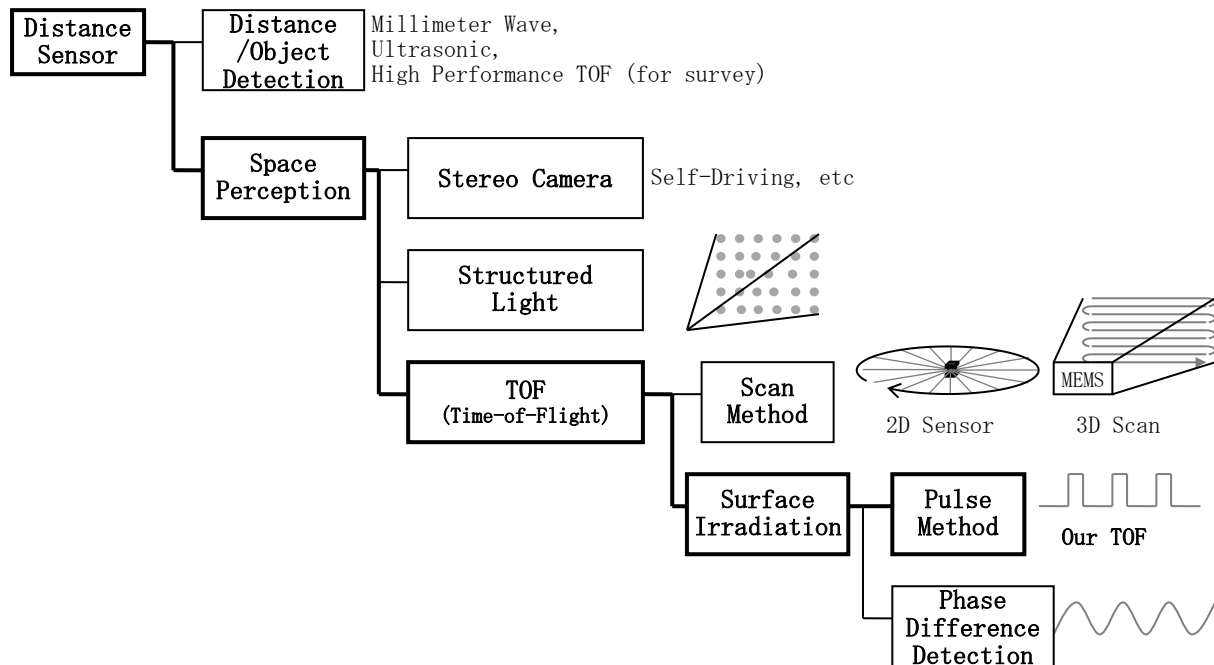


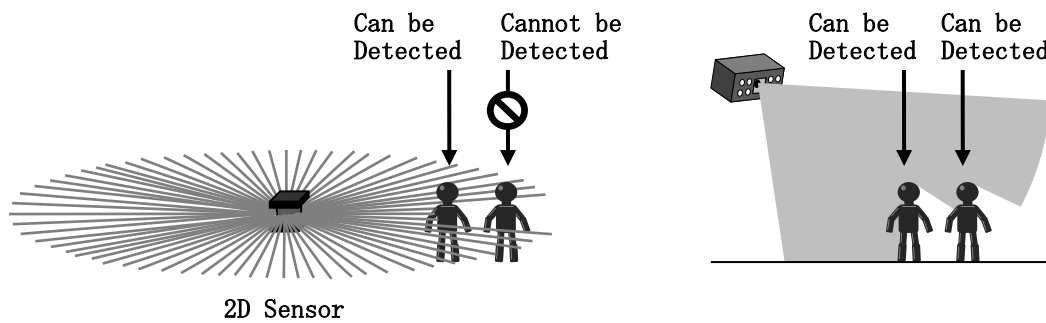
Figure 3 Types of Distance Sensor

Table 2-1 shows characteristics and differences of each distance sensor. "Surface irradiation method" which the TOF sensor of Hitachi-LG Data Storage uses can measure many points precisely in a space during short period. Since distances between all points are measured by hardware and does not require complicated calculation by software, it is an advantage that CPU work load of sensor itself and PC/Server connected with sensors is low.

Table 2-1 Comparison of Distance Sensors

Method		TOF		Structured Light	Stereo Camera
		Surface Irradiation	Scan Method		
Light Source		IR(Sine wave/Triangle wave/Pulse)		IR Special Pattern (Continuous/Pulse)	None (w/o assist light)
Measure Performance	Scope	Short	Long (✓)	Short	Long (✓)
	Accuracy	High (✓)	High (✓)	Middle	Middle
	Speed	Fast (✓)	Slow	Slow	Slow
	Points	Many (✓)	Few	Few	Many (✓)
Environment	Outdoor	Weak	Strong (✓)	Weak	Strong (✓)
	Dark Place	Strong (✓)	Strong (✓)	Strong (✓)	Weak
Caluculation Amount (CPU load)		Small (✓)	Small (✓)	Big	Big

2D sensor has been often used for human tracking and human sensor. General 2D sensor using laser measures distances between objects around the sensor rotating emission part and receiving part 360 degrees. It can measure precisely with the simple mechanism however it is a weak point that points over the object cannot be detected since it can measure only 2D in horizontal direction. For example, if several people are lining in the front and back, only the most front person can be detected. On the other hand, 3D sensor also has blind spot for measurement, all people can be detected adjusting sensor position and angle although multiple people are lined up. (Refer to Figure 4)



**Figure 4 2D Sensor and 3D Sensor**

It is expected that 3D TOF sensor will be used in many use cases.

## 3 Hardware

### 3.1 Hardware Specification

Table 3-1 shows the hardware specification of Hitachi-LG Data Storage's TOF sensor which this software supports.

**Table 3-1 TOF Sensor Hardware Specification**

Product Name	Motion Sensor		
Model Type	HLS-LFOM1 	HLS-LFOM3 (IP66) 	HLS-LFOM5 (Ceiling type) 
Light Source	Laser Diode $\lambda=850\text{nm}$ 、Laser Safety IEC60825-1 class1		
Power Supply	Power over Ethernet Plus(PoE+) : IEEE802.3at		
Power Consumption	15W		
Detect Distance & Detector	Detect Distance : $0.7^{*1}$ to 10m (White Kent Paper) $^{*2}$		
Resolution & Accuracy	$\sigma \leq 80\text{mm}$ @2m 320 x 240 pixel worth Measurement Accuracy : under 8% $^{*3}$ Ambient Illumination : under 10000lx $^{*2}$ : (White Kent Paper) * Measured at FOV center		
Field of View (FOV)	$H76^{\circ} \times V60^{\circ}$		$H60^{\circ} \times V76^{\circ}$
Image Resolution	TOFv1: Max 640 × 480 TOFv2: Max 320 × 240		TOFv1: Max 480 × 640 TOFv2: Max 240 × 320
Frame Rate	Max 30fps		
Interface	Ethernet 100 BASE-TX		
Indicator Light	Power Indicator Light (Green) : Lighting during power on Operation Indicator Light (Orange) : Flashing at normal, Turn off at out of order		
Ambient Temperature & Humidity	Guaranteed in 0 degree C to +45 degree C (Operational in 50 degree C) 0 to 95% (Non-condensing)		
Storage Temperature	-40 to +60 degree C		
Life	22,500hours		
Weight (Except cable)	540g	800g	500g
Dimensions (Except projecting part)	W138・D69・H69mm	W164・D73・H83mm	W150・D148・H44mm

Remark)  $^{*1}$  It may be saturated depending on target object. Please confirm ambient impact in the real environment.

$^{*2}$  Indoor environment (Fluorescent light under 1000lx)

$^{*3}$  Please confirm ambient impact in the real environment.

## 3.2 Hardware Types

There are basically 2 types of hardware in Hitachi-LG Data Storage's TOF sensor series (HLS-LFOM1/HLS-LFOM3/HLS-LFOM5) depending on firmware and FPGA. Those types are called as "TOFv1" and "TOFv2" in this document for convenience. Differences of each type are in Table 3-2. TOFv1 and TOFv2 can be switched by latest firmware (after FW ver.0100 of HLS-LFOM1/3, and all of HLS-LFOM5)(Refer to #4.5.3). The current hardware type can be confirmed by `Tof::tofinfo.tofver` (Refer to #6.9.1). Constrains of conventions of hardware and SDK are in #4.3.1. TOFv1 sensor can be updated to TOFv2 sensor. If you want it, please ask to Hitachi-LG Data Storage.

**Table 3-2 Hardware types**

Functions	TOFv1	TOFv2	
	HLS-LFOM1 HLS-LFOM3 HLS-LFOM5	HLS-LFOM1 HLS-LFOM3	HLS-LFOM5 *1
Depth resolution	16bit (1/65535)	8bit (1/255)	
Max image resolution	640×480 (VGA)	320×240 (QVGA)	
Noise reduction (Frame average, Median filter)	No	Yes	
Output depth and IR simultaneously	No	Yes	
Background subtraction	No	Yes	
Web GUI	No	Yes*3	Yes
Upside down mode	No	No	Yes *2

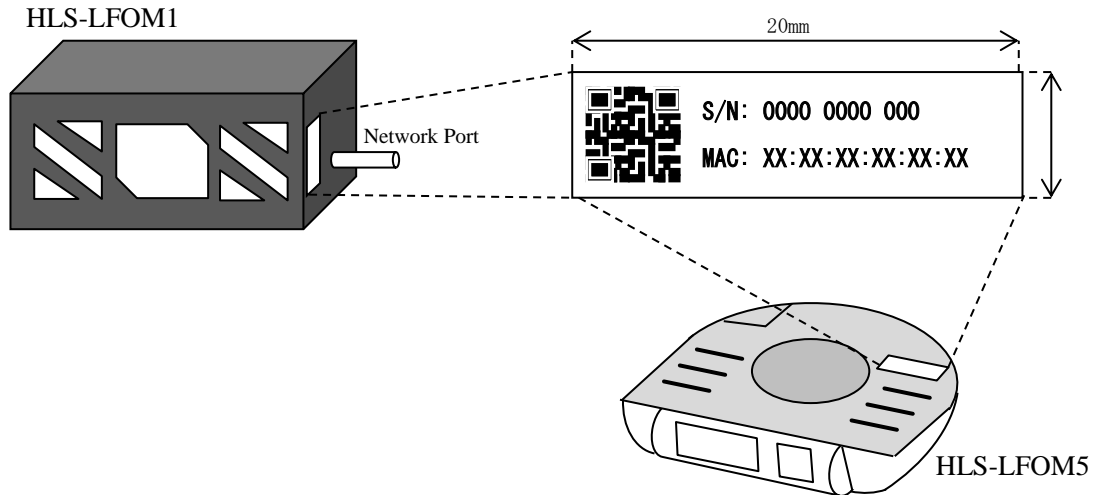
\*1 : It will be applied to HLS-LFOM1/2 later.

\*2 : Available after FW Ver.0100.

\*3 : Available after FW Ver.0500.

### 3.3 TOF ID

This software uses TOF ID to manage individual TOF sensor. TOF ID must be unique and immutable value, MAC address is used. MAC address is printed on QR code label (Refer to Figure 5) of the right side (Network port side) on HLS-LFOM1, and backside on HLS-LFOM5.



**Figure 5 QR Code**

Table 3-3 shows information included in QR Code. In such contents are printed in the QR Code label of Figure 5.

**Table 3-3 Contents of QR Code**

Item	Digits	Description	QR Code	Label
Production Date	1	Year : Last digit of A.D. (0 to 9)	✓	✓
	1	Month : 'A'(January) to 'L'(December)	✓	✓
	2	Day : (01 to 31)	✓	✓
Serial Number	3	Decimal	✓	✓
Managed Number	4	Decimal	✓	✓
MAC Address	12	ASCII	✓	✓

## 4 Getting Started

### 4.1 License

This SDK and the firmware in TOF sensor consist of several independent modules, and copyrights of our company and third parties exist in each module.

This SDK and TOF sensor include software modules that our company developed or implemented, such software and bundled documents also have copyrights and property rights of our company. Those are protected by copyright or other law.

This SDK and the firmware in TOF sensor consist of the following software.

- Freeware which has copyrights and property rights of third parties.
- Software permitted to use under GNU General Public License (GPL).
- Software permitted to use under GNU Lesser General Public License (LGPL)
- Software which is licensed by Open Market, Inc.
- Software permitted to use under BSD License.

Please do not ask us about contents of such open source code. Source code which have copyrights and property rights of our company or third parties are not delivered.

Applicable Software Module	Related Software License
Debian GNU/Linux6.0("squeeze")	Debian Free Software Guidelines
FFmpeg	GNU Lesser General Public License (LGPL) version 2.1
LIVE555 Streaming Media	GNU Lesser General Public License (LGPL) Version 2.1
Fast CGI	Open Market, Inc.
OpenCV	BSD License
libcurl	MIT License



## 4.2 System Requirement

### 4.2.1 Development Environment

Table 4-1 shows development environment for user application which uses this SDK.

**Table 4-1 Development Environment**

Hardware(PC) Requirements	1.6GHz or more processor 4GB or more memory
OS	Windows7/8/8.1/10 (x86 and x64)
	Ubuntu 14.04 LTS (x64)
	Ubuntu 16.04 LTS (x64)
	Debian 8.0 (x64)
	CentOS 7.0 (x64)
Development Environment	Visual Studio Community 2013 Visual Studio Ultimate 2013 Visual Studio Premium 2013 Visual Studio Professional 2013 Visual Studio Express 2013 for Desktop
	Visual Studio Community 2015 Visual Studio Professional 2015 Visual Studio Enterprise 2015
	gcc, g++ (-std=gnu++11 option mandatory) make cmake pkg-config
Development Language	VC++/C++

### 4.2.2 Operational Environment

Table 4-2 shows operational environment for user application which uses this SDK. Performance based on the hardware requirement depends on number of TOF sensors, hardware resource usage of user application.

**Table 4-2 Operational Environment**

Hardware (PC) Requirements	1.6GHz or more processor 4GB or more memory 1 port or more NIC (100BASE-TX/1000BASE-T)
OS	Windows7/8/8.1/10 (x86 and x64)
	.NET Framework
	Ubuntu 14.04 LTS (x64)
	Ubuntu 16.04 LTS (x64)
	Debian 8.0 (x64)
	CentOS 7.0 (x64)

## 4.3 Compatibility

### 4.3.1 Hardware Compatibility

Constraints of operation depending on conventions between hardware types of TOF sensor (Refer to #3.2) and SDK version are shown in Table 4-3. It is recommended to use the latest SDK version to support all type of hardware.

**Table 4-3 Hardware Compatibility**

Model	Firmware/FPGA	Hardware Type	SDK バージョン			
			v1.X.X	v2.0.X	v2.1.X	v2.2.X
HLS-LFOM1 HLS-LFOM3	FW:Ver.0006 or later FPGA:Ver.0002 or later	TOFv1 (Unchangeable)	Operational	Operational	Operational	Operational
	FW:Ver.0009 or later FPGA:Ver.0401 or later	TOFv2 (Unchangeable)	Not Operational	Operational	Operational	Operational
	FW:Ver.0200 or later FPGA:Ver.0401 or later	TOFv1	Not Operational	Not Operational	Operational	Operational
		TOFv2	Not Operational	Operational	Operational	Operational
HLS-LFOM5	FW:Ver.0100 or later FPGA:Ver.0507 or later	TOFv1	Not Supportyt	Not Supportyt	Not Supportyt	Operational
		TOFv2	Not Supportyt	Not Supportyt	Not Supportyt	Operational

### 4.3.2 Software Compatibility (for Windows)

This SDK (for Windows) consists of Driver which is installed in application PC, and Static Library which is linked at building application and Dynamic Link Library (DLL), and developers can choose either one.

It is recommended that Driver version and Static Library version is same, however it is possible to execute old applications built with old Static Library (v1.X.X) if Driver is updated to the latest v2.X.X (Refer to Table 4-4). However applications built with Static Library v1.X.X cannot operate TOFv2 sensors.

**Table 4-4 Software Compatibility of Static Library Driver (tof.lib)**

		Static Library			Remark
		v1.X.X	v2.0.X	v2.1.X or later	
Driver	v1.X.X	Operational (Only TOFv1)	Not Operational	Not Operational	
	v2.0.X	Operational (Only TOFv1)	Operational	Operational [*1]	
	v2.1.X or later	Operational (Only TOFv1)	Operational [*2]	Operational	

\*1 : Put tof14.dll(tof14d.dll) and opencv\_world310.dll(opencv\_world310d.dll) in a folder together with the application which uses SDK for Visual Studio 2015.

\*2 : Setting of TOFv2 PicByte = 2 (support VGA) is not supported.

DLL is supported from v2.1.x. As Table 4-5, same version of DLL and driver shall be used. Some bug fixes are not applied if driver of v2.0.X is used.

**Table 4-5 Software Compatibility of DLL (tof.dll.lib, tof12.dll, tof14.dll)**

		Dynamic Library			Remark
		v1.X.X	v2.0.X	v2.1.X or later	
Driver	v1.X.X	n/a	n/a	Not Operational	
	v2.0.X	n/a	n/a	Not Operational (Conditionally operationlan[*1][*2])	
	v2.1.X or later	n/a	n/a	Operational	

\*1 : Put tof12.dll(tof12d.dll) in a folder together with the application which uses SDK for Visual Studio 2013.

\*2 : Put tof14.dll(tof14d.dll) and opencv world310.dll(opencv world310d.dll) in a folder together with the application which uses SDK for Visual Studio 2015.

Table 4-6 shows compatibly between static library and DLL consisting v2.1.x, and OpenCV 2.4.x/3.1.0.

**Table 4-6 Compatibility with OpenCV**

		Dynamic Library				Remark
		VS2013		VS2015		
		Static Library (tof.lib)	DLL (tof12.dll, tofdll.lib)	Static Library (tof.lib)	DLL (tof14.dll, tofdll.lib)	
OpenCV	v2.4.10 v2.4.11 v2.4.13	Operational	Operational	Cannot build	Operational [*1]	
	v3.1.0	Cannot build	Operational [*2]	Operational [*2]	Operational [*2]	

\*1 : Since binary of v2.4.x is not published from OpenCV official site, users must build source code.

\*2 : Binary of v3.1.0 32bit is not published from OpenCV official site.

## 4.4 Setup Development Environment

### 4.4.1 Windows7/8/8.1/10

#### 4.4.1.1 Distributions [Windows7/8/8.1/10 VS2013/VS2015]

SDK for Windows Visual Studio 2013 VC++ includes distributions listed in Table 4-7. Those materials must be installed according to #4.4.1.2 and #4.4.1.3.

**Table 4-7 Distributions for Visual Studio 2013 VC++ (HldsTofSdk.2.2.0vs2013 folder)**

Folder			File	Contents
/x86	/tofsdk	/include	tof.h	Header file
		/lib	tof.lib	Static link library(for 32bit OS/Release mode)
			tofd.lib	Static link library(for 32bit OS/Debug mode)
		/dll	tof12.dll	Dynamic link library(for 32bit OS/Release mode)
			tofdll.lib	Import library for DLL (for 32bit OS/Release mode)
			tof12d.dll	Dynamic link library(for 32bit OS/Debug mode)
			tofdlld.lib	Import library for DLL (for 32bit OS/Debug mode)
	/tofdriver		tof_driver_x86_v2.2.0_Installer.exe	Driver Installer (for 32bit OS)
/x64	/tofsdk	/include	tof.h	Header file
		/lib	tof.lib	Static link library(for 64bit OS/Release mode)
			tofd.lib	Static link library(for 64bit OS/Debug mode)
		/dll	tof12.dll	Dynamic link library(for 64bit OS/Release mode)
			tofdll.lib	Import library for DLL (for 64bit OS/Release mode)
			tof12d.dll	Dynamic link library(for 64bit OS/Debug mode)
			tofdlld.lib	Import library for DLL (for 64bit OS/Debug mode)
	/tofdriver		tof_driver_x64_v2.2.0_Installer.exe	Driver Installer (for 64bit OS)
/sample		tof.ini	Sample of initialization file	
		Tof2dViewer.cpp	Sample programs (source code)	
		Tof3dViewer_cv.cpp		
		TofIrViewer.cpp		
		Tofv2Viewer.cpp		
		HumanCounter.cpp		
		/x86	Tof2dViewer.exe	Sample execution files (for 32bit OS) * It is needed to install PCL for Tof3dViewer_pcl.exe.
			Tof3dViewer_cv.exe	
			Tof3dViewer_pcl.exe	
			TofIrViewer.exe	
			Tofv2Viewer.exe	
			HumanCounter.exe	
			tof.ini	Sample of initialization file
		opencv_core2410.dll	Necessary .DLL files to execute sample programs. (for 32bit OS)	
				opencv_highgui2410.dll
				opencv_imgproc2410.dll
/x64	Tof2dViewer.exe	Sample execution files (for 64bit OS) * It is needed to install PCL for Tof3dViewer_pcl.exe.		
	Tof3dViewer_cv.exe			
	Tof3dViewer_pcl.exe			
	TofIrViewer.exe			
	Tofv2Viewer.exe			

		HumanCounter.exe	
		tof.ini	Sample of initialization file
		opencv_core2410.dll	Necessary .DLL files to execute sample programs. (for 64bit OS)
		opencv_highgui2410.dll	
		opencv_imgproc2410.dll	
/manual		HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual
/utility		tof_utility.exe	Utility tool

SDK for Windows Visual Studio 2015 VC++ includes distributions listed in Table 4-8. Those materials must be installed according to #4.4.1.2 and #4.4.1.3.

**Table 4-8 Distributions for Visual Studio 2015 VC++ (HldsTofSdk.2.2.0vs2015 folder)**

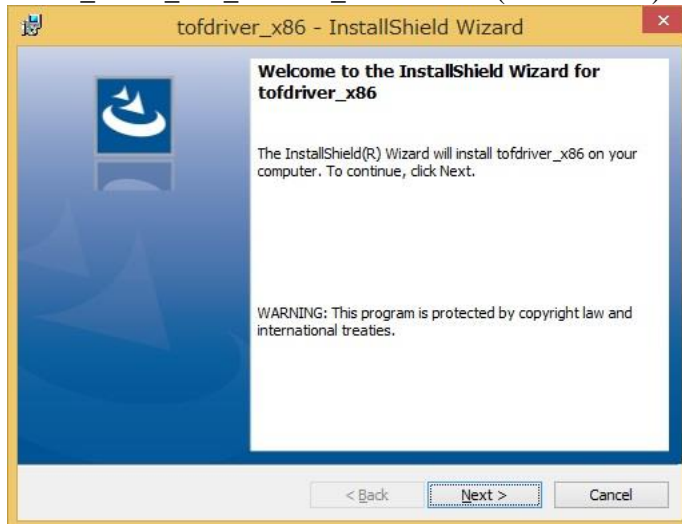
Folder			File	Contents
/x86	/tofsdk	/include	tof.h	Header file
		/lib	tof.lib	Static link library(for 32bit OS/Release mode)
			tofd.lib	Static link library(for 32bit OS/Debug mode)
		/dll	tof14.dll	Dynamic link library(for 32bit OS/Release mode)
			tofdll.lib	Import library for DLL (for 32bit OS/Release mode)
			tof14d.dll	Dynamic link library(for 32bit OS/Debug mode)
			tofdlld.lib	Import library for DLL (for 32bit OS/Debug mode)
	/tofdriver		tof_driver_x86_v2.2.0_Installer.exe	Driver Installer (for 32bit OS)
/x64	/tofsdk	/include	tof.h	Header file
		/lib	tof.lib	Static link library(for 64bit OS/Release mode)
			tofd.lib	Static link library(for 64bit OS/Debug mode)
		/dll	tof14.dll	Dynamic link library(for 64bit OS/Release mode)
			tofdll.lib	Import library for DLL (for 64bit OS/Release mode)
			tof14d.dll	Dynamic link library(for 64bit OS/Debug mode)
			tofdlld.lib	Import library for DLL (for 64bit OS/Debug mode)
	/tofdriver		tof_driver_x64_v2.2.0_Installer.exe	Driver Installer (for 64bit OS)
/sample		tof.ini	Sample of initialization file	
		Tof2dViewer.cpp	Sample programs (source code)	
		Tof3dViewer_cv.cpp		
		TofIrViewer.cpp		
		Tofv2Viewer.cpp		
		HumanCounter.cpp		
		/x86	Tof2dViewer.exe	Sample execution files (for 32bit OS) * It is needed to install PCL for Tof3dViewer_pcl.exe.
			Tof3dViewer_cv.exe	
			Tof3dViewer_pcl.exe	
			TofIrViewer.exe	
			Tofv2Viewer.exe	
			HumanCounter.exe	
		/x86	tof.ini	Sample of initialization file
tof14.dll	Necessary .DLL files to execute sample programs. (for 32bit OS)			
opencv_world310.dll				
/x64	Tof2dViewer.exe		Sample execution files (for 64bit OS)	
	Tof3dViewer_cv.exe			

		Tof3dViewer_pcl.exe	* It is needed to install PCL for Tof3dViewer_pcl.exe.
		TofIrViewer.exe	
		Tofv2Viewer.exe	
		HumanCounter.exe	
		tof.ini	Sample of initialization file
		tof14.dll	Necessary .DLL files to execute sample programs. (for 64bit OS)
		opencv_world310.dll	
/manual		HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual
/utility		tof_utility.exe	Utility tool

#### 4.4.1.2 Installing driver (for Users/Developers) [Windows7/8/8.1/10]

Execute an installer of x86(for 32bit OS) or x64(for 64bit OS) in /tofdriver folder of distribution depending on PC which user application runs. This process is necessary for both of user application operational environment and user application development environment.

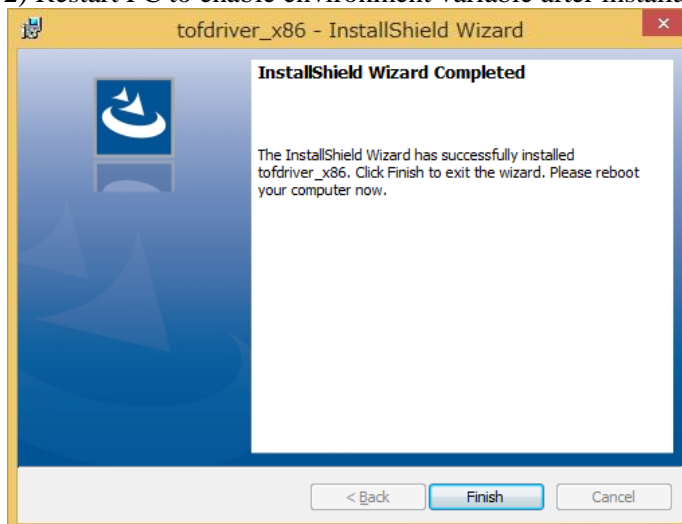
- 1) Install TOF driver x86(for 32bit OS) or x64(for 64bit OS)  
tof\_driver\_x86\_vX.X.X\_Installer.exe (for 32bit OS)  
tof\_driver\_x64\_vX.X.X\_Installer.exe (for 64bit OS)



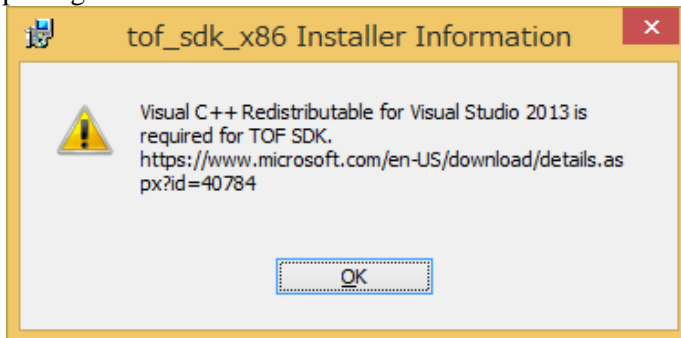
Installer does the following items.

- Copy necessary files (driver and so on) to “C:/Program Files/HLDS/TofSdk/tofdriver”.
- Add “TOFSDKPATH” to the environment variable and set above path.
- Set above path to the environment variable “PATH” also.

- 2) Restart PC to enable environment variable after installation.



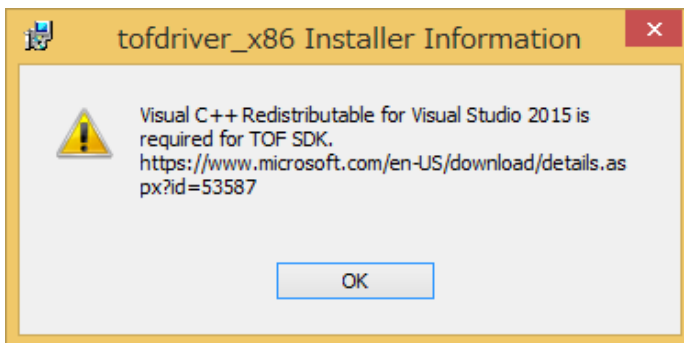
3) If the following error occurred during installation, Visual Studio 2013 Visual C++ redistributable package is needed.



Please download and install Visual Studio 2013 Visual C++ redistributable package from the following Microsoft site.

<https://www.microsoft.com/en-US/download/details.aspx?id=40784>

4) If the following error occurred during installation, Visual Studio 2015 Visual C++ redistributable package is needed.



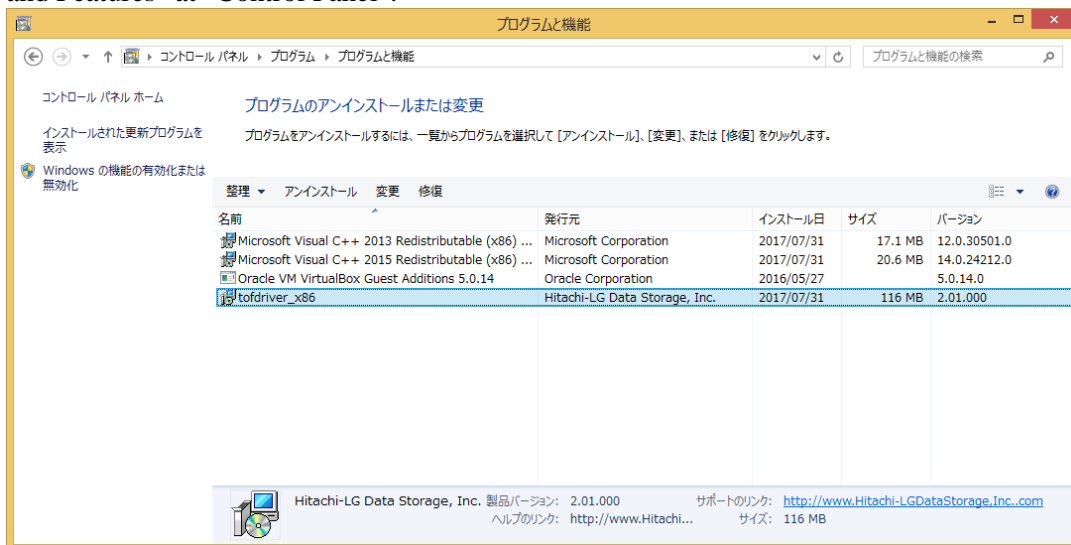
Please download and install Visual Studio 2015 Visual C++ redistributable package from the following Microsoft site.

<https://www.microsoft.com/en-US/download/details.aspx?id=53587>

\* Both of redistributable packages of VS2013 and VS2015 are required for v2.1.x or later.



5) When uninstalling the driver, please uninstall tofdriver\_x86 or tofdriver\_x64 from "Programs and Features" at "Control Panel".



If copied folder in (1) and environment variable are not deleted, delete them manually.

#### 4.4.1.3 Installing SDK (for Developers only) [Windows7/8/8.1/10]

/tofsdk folder in distribution must be copied to development environment which user application is developed. x86(for 32bit OS) or x64(for 64bit OS) must be selected depending on OS which user application runs.

##### To develop with Visual Studio 2013 Static link library (tof.lib, tofd.lib)

- 1) Make a new project for user application on Visual Studio 2013.
- 2) Copy /tofsdk under HldsTofSdk.X.X.Xvs2013/x86 or HldsTofSdk.X.X.Xvs2013/x64 to any location under the project folder. (ex: [Project Folder]/packages/tofsdk)
- 3) Add /include folder of 2) to [Include Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/include)  
Add /lib folder of 2) to [Library Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/lib)
- 4) Add tof.lib (for Release mode) or tofd.lib (for Debug mode) to [Additional Dependencies] of [Configuration Properties] - [Linker] - [Input] of Project Properties.

##### To develop with Visual Studio 2013 DLL (tof12.dll/tofdll.lib, tof12d.dll/tofdlld.lib)

- 1) Make a new project for user application on Visual Studio 2013.
- 2) Copy /tofsdk under HldsTofSdk.X.X.Xvs2013/x86 or HldsTofSdk.X.X.Xvs2013/x64 to any location under the project folder. (ex: [Project Folder]/packages/tofsdk)
- 3) Add /include folder of 2) to [Include Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/include)  
Add /dll folder of 2) to [Library Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/dll)
- 4) Add tofdll.lib (for Release mode) or tofdlld.lib (for Debug mode) to [Additional Dependencies] of [Configuration Properties] - [Linker] - [Input] of Project Properties.

**To develop with Visual Studio 2015 Static link library (tof.lib, tofd.lib)**

- 1) Make a new project for user application on Visual Studio 2015.
- 2) Copy /tofsdk under HldsTofSdk.X.X.Xvs2015/x86 or HldsTofSdk.X.X.Xvs2015/x64 to any location under the project folder. (ex: [Project Folder]/packages/tofsdk)
- 3) Add /include folder of 2) to [Include Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/include)  
Add /lib folder of 2) to [Library Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/lib)
- 4) Add tof.lib (for Release mode) or tofd.lib (for Debug mode) to [Additional Dependencies] of [Configuration Properties] - [Linker] - [Input] of Project Properties.

**To develop with Visual Studio 2015 DLL (tof14.dll/tofdll.lib, tof14d.dll/tofdlld.lib)**

- 1) Make a new project for user application on Visual Studio 2015.
- 2) Copy /tofsdk under HldsTofSdk.X.X.Xvs2015/x86 or HldsTofSdk.X.X.Xvs2015/x64 to any location under the project folder. (ex: [Project Folder]/packages/tofsdk)
- 3) Add /include folder of 2) to [Include Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/include)  
Add /dll folder of 2) to [Library Directories] of [Configuration Properties] - [Project Master Settings] - [VC++ Directories] in Project Properties.  
(ex:\$(SolutionDir)packages/tofsdk/dll)
- 4) Add tofdll.lib (for Release mode) or tofdlld.lib (for Debug mode) to [Additional Dependencies] of [Configuration Properties] - [Linker] - [Input] of Project Properties.

**4.4.1.4 Build(for Developers only) [Windows7/8/8.1/10]**

- 1) Make Win32 console application project. At the “Application Settings” of “Win32 Application Wizard”, check the box of the “Empty project” option.
- 2) Add Sample programs source code to the project.
- 3) Install this SDK as #4.4.
- 4) Install OpenCV.

## 4.4.2 Ubuntu14.04

### 4.4.2.1 Distributions [Ubuntu14.04]

This SDK includes distributions listed in Table 4-9. Those materials must be installed according to #4.4.2.2.

**Table 4-9 Distributions [Ubuntu 14.04]**

Folder	File	Contents
/HldsTofSdk.2.2.0ubuntu14_x64	libtof-dev_2.2.0-1ubuntu14_amd64.deb	libtof-dev debian package for 64bit
/manual	HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual

File list of libtof-dev\_2.2.0-1ubuntu14\_amd64.deb.

Folder		File	Contents
/usr/share/doc/libtof-dev/		copyright	copyright
		README.Debian	README
		changelog.Debian.gz	changelog
/usr/include/hldstof/		tof.h	Header file
/usr/lib/hldstof/		libtof.a	Static Library
/usr/share/doc/libtof-dev/examples/	TofImageCapture/	TofImage.cpp.gz	Sample program (not use OpenCV)
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof2dViewer/	Tof2dViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof3dViewer_cv/	Tof3dViewer_cv.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	TofIrViewer/	TofIrViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	HumanCounter/	HumanCounter.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
		HumanCounter.ini	HumanCounter setting file
	Tofv2Viewer/	Tofv2Viewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file

\*1 : need OpenCV Developer Package

#### 4.4.2.2 Installing driver (for Users/Developers) [Ubuntu14.04]

1) Install developer's tools.

```
$ sudo apt-get install gcc g++ make cmake pkg-config
```

If building sample code which needs OpenCV, it needs to install libopencv-dev or build it from OpenCV source.

```
$ sudo apt-get install libopencv-dev
```

2) Install TOF SDK.

```
$ sudo dpkg -i libtof-dev_2.2.0-1ubuntu14_amd64.deb
Selecting previously unselected package libtof-dev.
(Reading database ... ***** files and directories currently installed.)
Preparing to unpack libtof-dev_2.2.0-1ubuntu14_amd64.deb ...
Unpacking libtof-dev (2.2.0-1ubuntu14) ...
Setting up libtof-dev (2.2.0-1ubuntu14) ...
Processing triggers for libc-bin (2.19-0ubuntu6) ...
```

3) Build with libtof.a

g++ example (no OpenCV sample : TofImage.cpp)

```
$ g++ -std=gnu++11 -o TofImage.o -c TofImage.cpp
$ g++ -std=gnu++11 TofImage.o -o TofImageCapture /usr/lib/hldstof/libtof.a -lrt -lm -lpthread
-lstdc++
```

g++ example (OpenCV sample : Tof2dViewer.cpp)

```
$ g++ -std=gnu++11 `pkg-config --cflags opencv` -o Tof2dViewer.o -c Tof2dViewer.cpp
$ g++ -std=gnu++11 Tof2dViewer.o -o Tof2dViewer `pkg-config --static --libs opencv`
/usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

Remark : ` : Accent grave key

4) Build with libtof.a using CMakeLists.txt

Copy sample code.

```
$ cp -ra /usr/share/doc/libtof-dev/examples/ .
$ cd examples
```

Unpacking cpp.gz, Create build folder.

```
$ cd <sample folder>
$ gunzip *.cpp.gz
$ mkdir build
$ cd build/
```

Create Makefile using cmake, Execute Sample.

```
$ cmake ../
$ make
$ cp ../tof.ini .
$ vi tof.ini                                # modify initialization file for your TOF IP address.
$ ./<Sample name>
```

CMakeLists.txt example (no OpenCV sample : TofImage.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

add_executable(TofImageCapture TofImage.cpp)
target_link_libraries(TofImageCapture /usr/lib/hldstof/libtof.a rt m pthread stdc++)
```

CMakeLists.txt example (OpenCV sample : Tof2dViewer.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

find_package(PkgConfig)
pkg_check_modules(LIBOPENCV2 REQUIRED opencv)
include_directories(${LIBOPENCV2_INCLUDE_DIRS})

add_executable(Tof2dViewer
    Tof2dViewer.cpp)
target_link_libraries(Tof2dViewer
    ${LIBOPENCV2_LDFLAGS} ${LIBOPENCV2_LIBRARIES} /usr/lib/hldstof/libtof.a rt
    m pthread stdc++)
```

## 4.4.3 Ubuntu 16.04

### 4.4.3.1 Distributions [Ubuntu 16.04]

This SDK includes distributions listed in Table 4-10. Those materials must be installed according to #4.4.3.2.

**Table 4-10 Distributions [Ubuntu 16.04]**

Folder	File	Contents
HldsTofSdk.2.2.0ubuntu16_x64	libtof-dev_2.2.0-1ubuntu16_amd64.deb	libtof-dev debian package for 64bit
/manual	HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual

File list of libtof-dev\_2.2.0-1ubuntu16\_amd64.deb

Folder		File	Contents
/usr/share/doc/libtof-dev/		copyright	copyright
		README.Debian	README
		changelog.Debian.gz	changelog
/usr/include/hldstof/		tof.h	Header file
/usr/lib/hldstof/		libtof.a	Static Library
/usr/share/doc/libtof-dev/examples/	TofImageCapture/	TofImage.cpp.gz	Sample program (not use OpenCV)
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof2dViewer/	Tof2dViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof3dViewer_cv/	Tof3dViewer_cv.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	TofIrViewer/	TofIrViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	HumanCounter/	HumanCounter.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
		HumanCounter.ini	HumanCounter setting file
	Tofv2Viewer/	Tofv2Viewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file

\*1 : need OpenCV Developer Package

#### 4.4.3.2 Installing driver (for Users/Developers) [Ubuntu 16.04]

1) Install developer's tools.

```
$ sudo apt-get install gcc g++ make cmake pkg-config
```

If building sample code which needs OpenCV, it needs to install libopencv-dev or build it from OpenCV source.

```
$ sudo apt-get install libopencv-dev
```

2) Install TOF SDK.

```
$ sudo dpkg -i libtof-dev_2.2.0-1ubuntu16_amd64.deb
Selecting previously unselected package libtof-dev.
(Reading database ... ***** files and directories currently installed.)
Preparing to unpack libtof-dev_2.2.0-1ubuntu16_amd64.deb ...
Unpacking libtof-dev (2.2.0-1ubuntu16) ...
Setting up libtof-dev (2.2.0-1ubuntu16) ...
```

3) Build with libtof.a

g++ example (no OpenCV sample : TofImage.cpp)

```
$ g++ -std=gnu++11 -o TofImage.o -c TofImage.cpp
$ g++ -std=gnu++11 TofImage.o -o TofImageCapture /usr/lib/hldstof/libtof.a -lrt -lm -lpthread
-lstdc++
```

g++ example (OpenCV sample : Tof2dViewer.cpp)

```
$ g++ -std=gnu++11 `pkg-config --cflags opencv` -o Tof2dViewer.o -c Tof2dViewer.cpp
$ g++ -std=gnu++11 Tof2dViewer.o -o Tof2dViewer `pkg-config --static --libs opencv`
/usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

Remark : ` : Accent grave key

4) Build with libtof.a using CMakeLists.txt

Copy sample code.

```
$ cp -ra /usr/share/doc/libtof-dev/examples/ .
$ cd examples
```

Unpacking cpp.gz, Create build folder.

```
$ cd <sample folder>
$ gunzip *.cpp.gz
$ mkdir build
$ cd build/
```

Create Makefile using cmake, Execute Sample.

```
$ cmake ../
$ make
$ cp ../tof.ini .
$ vi tof.ini                                # modify initialization file for your TOF IP address.
$ ./<Sample name>
```

CMakeLists.txt example (no OpenCV sample : TofImage.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

add_executable(TofImageCapture TofImage.cpp)
target_link_libraries(TofImageCapture /usr/lib/hldstof/libtof.a rt m pthread stdc++)
```

CMakeLists.txt example (OpenCV sample : Tof2dViewer.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

find_package(PkgConfig)
pkg_check_modules(LIBOPENCV2 REQUIRED opencv)
include_directories(${LIBOPENCV2_INCLUDE_DIRS})

add_executable(Tof2dViewer
    Tof2dViewer.cpp)
target_link_libraries(Tof2dViewer
    ${LIBOPENCV2_LDFLAGS} ${LIBOPENCV2_LIBRARIES} /usr/lib/hldstof/libtof.a rt
    m pthread stdc++)
```



## 4.4.4 Debian 8.0

### 4.4.4.1 Distributions [Debian 8.0]

This SDK includes distributions listed in Table 4-11. Those materials must be installed according to #4.4.4.2.

**Table 4-11 Distributions [Debian 8.0]**

Folder	File	Contents
HldsTofSdk.2.2.0debian8_x64	libtof-dev_2.2.0-1debian8_amd64.deb	libtof-dev debian package for 64bit
/manual	HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual

File list of libtof-dev\_2.2.0-1debian8\_amd64.deb

Folder		File	Contents
/usr/share/doc/libtof-dev/		copyright	copyright
		README.Debian	README
		changelog.Debian.gz	changelog
/usr/include/hldstof/		tof.h	Header file
/usr/lib/hldstof/		libtof.a	Static Library
/usr/share/doc/libtof-dev/examples/	TofImageCapture/	TofImage.cpp.gz	Sample program (not use OpenCV)
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof2dViewer/	Tof2dViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof3dViewer_cv/	Tof3dViewer_cv.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	TofIrViewer/	TofIrViewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	HumanCounter/	HumanCounter.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
		HumanCounter.ini	HumanCounter setting file
	Tofv2Viewer/	Tofv2Viewer.cpp.gz	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file

\*1 : need OpenCV Developer Package

#### 4.4.4.2 Installing driver (for Users/Developers) [Debian 8.0]

##### 1) Install developer's tools.

Since sudo privilege for general users is not created just after initial installation of Debian 8, next operation may be required.

Create sudo privileges for general users

```
$ su -  
root # apt-get install sudo  
root # visudo
```

Usage of visudo

- (1) Add following to # User privilege specification  
    <User name>      ALL=(ALL:ALL) ALL    # (ex) Assign same privileges as root
- (2) <ctrl> + "o"
- (3) <return>
- (4) <ctrl> + "x"

```
root # exit  
$
```

```
$ sudo apt-get install gcc g++ make cmake pkg-config
```

In case of building sample code which requires OpenCV, install libopencv-dev or build OpenCV source code.

Just after initial installation of Debian 8, install repository of libopencv-dev is not registered yet, the repository shall be registered.

```
$ sudo vi /etc/apt/sources.list  
# Add the following 2 lines  
deb http://ftp.jp.debian.org/debian/ jessie main  
deb-src http://ftp.jp.debian.org/debian/ jessie main  
$ sudo apt-get update
```

Install libopencv-dev

```
$ sudo apt-get install libopencv-dev
```

## 2) Install TOF SDK.

```
sudo dpkg -i libtof-dev_2.2.0-1debian8_amd64.deb
Selecting previously unselected package libtof-dev.
(Reading database ... ***** files and directories currently installed.)
Preparing to unpack libtof-dev_2.2.0-1debian8_amd64.deb ...
Unpacking libtof-dev (2.2.0-1debian8) ...
Setting up libtof-dev (2.2.0-1debian8) ...
Processing triggers for libc-bin (2.19-18+deb8u6) ...
```

## 3) Build with libtof.a

g++ example (no OpenCV sample : TofImage.cpp)

```
$ g++ -std=gnu++11 -o TofImage.o -c TofImage.cpp
$ g++ -std=gnu++11 TofImage.o -o TofImageCapture /usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

g++ example (OpenCV sample : Tof2dViewer.cpp)

```
$ g++ -std=gnu++11 `pkg-config --cflags opencv` -o Tof2dViewer.o -c Tof2dViewer.cpp
$ g++ -std=gnu++11 Tof2dViewer.o -o Tof2dViewer `pkg-config --static --libs opencv` /usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

Remark : ` : Accent grave key

## 4) Build with libtof.a using CMakeLists.txt

Unpacking cpp.gz, Create build folder.

```
$ cp -ra /usr/share/doc/libtof-dev/examples/ .
$ cd examples
```

Create Makefile using cmake, Execute Sample.

```
$ cd <sample folder>
$ gunzip *.cpp.gz
$ mkdir build
$ cd build/
```

Create Makefile using cmake, Execute Sample.

```
$ cmake ../
$ make
$ cp ../tof.ini .
$ vi tof.ini # modify initialization file for your TOF IP address.
$ ./<Sample name>
```

CMakeLists.txt example (no OpenCV sample : TofImage.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

add_executable(TofImageCapture TofImage.cpp)
target_link_libraries(TofImageCapture /usr/lib/hldstof/libtof.a rt m pthread stdc++)
```

CMakeLists.txt example (OpenCV sample : Tof2dViewer.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

find_package(PkgConfig)
pkg_check_modules(LIBOPENCV2 REQUIRED opencv)
include_directories(${LIBOPENCV2_INCLUDE_DIRS})

add_executable(Tof2dViewer
    Tof2dViewer.cpp)
target_link_libraries(Tof2dViewer
    ${LIBOPENCV2_LDFLAGS} ${LIBOPENCV2_LIBRARIES} /usr/lib/hldstof/libtof.a rt
    m pthread stdc++)
```

## 4.4.5 CentOS 7.0

### 4.4.5.1 Distributions [CentOS 7.0]

This SDK includes distributions listed in Table 4-12. Those materials must be installed according to #4.4.5.2.

**Table 4-12 Distributions [CentOS 7.0]**

Folder	File	Contents
HldsTofSdk.2.2.0centos7_x64	libtof-dev-2.2.0-1.el7.x86_64.rpm	libtof-dev rpm package for 64bit
/manual	HLDS_TOF_API_Reference_Manual_rX.X.pdf	This manual

File list of libtof-dev-2.2.0-1.el7.x86\_64.rpm

Folder		File	Contents
/usr/share/doc/libtof-dev/		copyright	copyright
/usr/include/hldstof/		tof.h	Header file
/usr/lib/hldstof/		libtof.a	Static Library
/usr/share/doc/libtof-dev/examples/	TofImageCapture/	TofImage.cpp	Sample program (not use OpenCV)
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof2dViewer/	Tof2dViewer.cpp	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	Tof3dViewer_cv/	Tof3dViewer_cv.cpp	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	TofIrViewer/	TofIrViewer.cpp	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
	HumanCounter/	HumanCounter.cpp	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file
		HumanCounter.ini	HumanCounter setting file
	Tofv2Viewer/	Tofv2Viewer.cpp	Sample program (use OpenCV [*1])
		CMakeLists.txt	for cmake
		tof.ini	Sample of initialization file

\*1 : need OpenCV Developer Package

#### 4.4.5.2 Installing driver (for Users/Developers) [CentOS 7.0]

1) Install developer's tools.

```
$ sudo yum install gcc gcc-c++ make cmake pkg-config
```

If building sample code which needs OpenCV, it needs to install opencv-devel or build it from OpenCV source.

```
$ sudo yum install opencv-devel
```

2) Install TOF SDK.

```
$ sudo rpm -Uvh libtof-dev-2.2.0-1.el7.x86_64.rpm
Preparing... ##### [100%]
Updating / installing...
 1:libtof-dev-2.2.0-1.el7 ##### [100%]
```

3) Build with libtof.a

g++ example (no OpenCV sample : TofImage.cpp)

```
$ g++ -std=gnu++11 -o TofImage.o -c TofImage.cpp
$ g++ -std=gnu++11 TofImage.o -o TofImageCapture /usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

g++ example (OpenCV sample : Tof2dViewer.cpp)

```
$ g++ -std=gnu++11 `pkg-config --cflags opencv` -o Tof2dViewer.o -c Tof2dViewer.cpp
$ g++ -std=gnu++11 Tof2dViewer.o -o Tof2dViewer `pkg-config --static --libs opencv` /usr/lib/hldstof/libtof.a -lrt -lm -lpthread -lstdc++
```

Remark : ` : Accent grave key

4) Build with libtof.a using CMakeLists.txt

Copy sample code.

```
$ cp -ra /usr/share/doc/libtof-dev/examples/ .
$ cd examples
```

Create build folder.

```
$ cd <sample folder>
$ mkdir build
$ cd build/
```

Create Makefile using cmake, Execute Sample.

```
$ cmake ../
$ make
$ cp ../tof.ini .
$ vi tof.ini                                # modify initialization file for your TOF IP address.
$ ./<Sample name>
```

CMakeLists.txt example (no OpenCV sample : TofImage.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

add_executable(TofImageCapture TofImage.cpp)
target_link_libraries(TofImageCapture /usr/lib/hldstof/libtof.a rt m pthread stdc++)
```

CMakeLists.txt example (OpenCV sample : Tof2dViewer.cpp)

```
set(CMAKE_C_COMPILER "gcc")

set(CMAKE_CXX_COMPILER "g++")
set(CMAKE_CXX_FLAGS "-std=gnu++11")

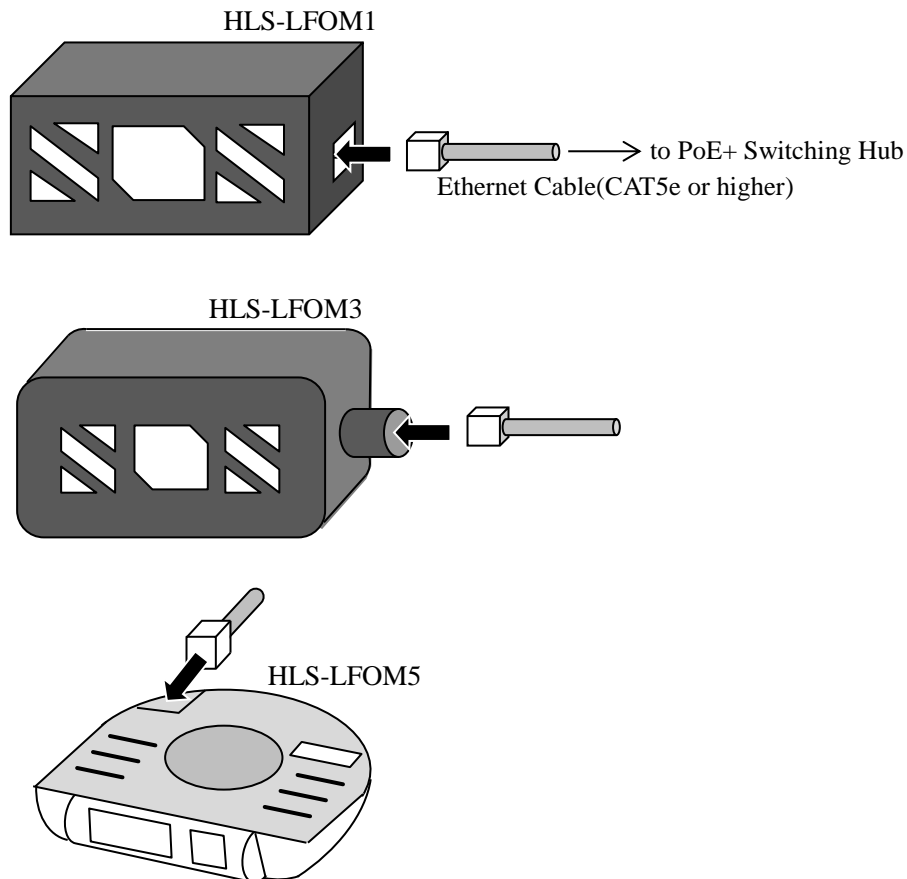
find_package(PkgConfig)
pkg_check_modules(LIBOPENCV2 REQUIRED opencv)
include_directories(${LIBOPENCV2_INCLUDE_DIRS})

add_executable(Tof2dViewer
    Tof2dViewer.cpp)
target_link_libraries(Tof2dViewer
    ${LIBOPENCV2_LDFLAGS} ${LIBOPENCV2_LIBRARIES} /usr/lib/hldstof/libtof.a rt
    m pthread stdc++)
```

## 4.5 Setup TOF Sensor

### 4.5.1 Setup and start

Hardware installation should be according to TOF Delivery Specification. When an Ethernet cable (CAT5e or higher) which power is supplied from PoE+ switching hub is connected to the TOF sensor as Figure 6, it is started. (No power switch).



**Figure 6 Cabling**

After started, TOF sensor is in Standby mode. When `Tof::Open()` method is called, Standby is canceled and lasers start to emit.

Disconnecting Ethernet cable powers off TOF sensor. However, `Tof::Close()` method must be called before disconnecting Ethernet cable if the TOF sensor is used by `Tof::Open()` method of `Tof` class.



## 4.5.2 Setting IP address

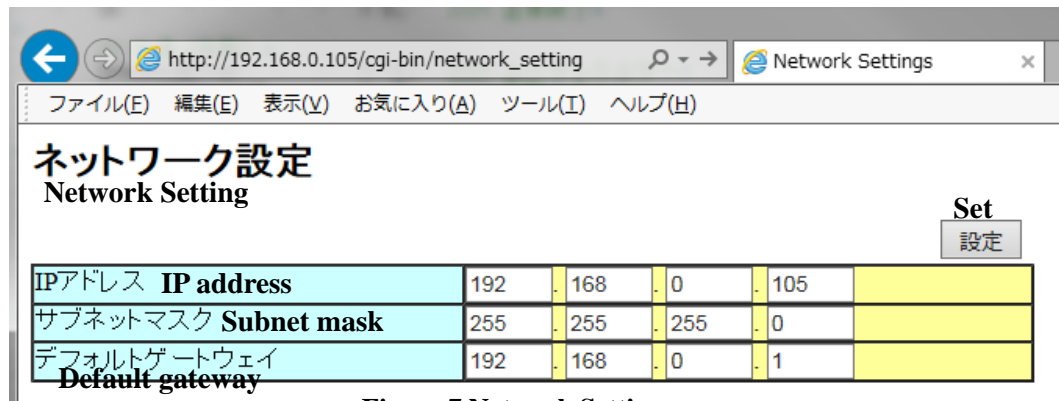
IP address must be set to TOF sensor before used by SDK. (IP address cannot be set by SDK.)

### 4.5.2.1 For models Web GUI is not available

Inputting the following URL to internet browser, network setting of Figure 7 is shown to set network.

**http://X.X.X.X/cgi-bin/network\_setting**

(X.X.X.X is IP address, and factory default is 192.168.0.105.)



IPアドレス	IP address	192	168	0	105	
サブネットマスク	Subnet mask	255	255	255	0	
デフォルトゲートウェイ	Default gateway	192	168	0	1	

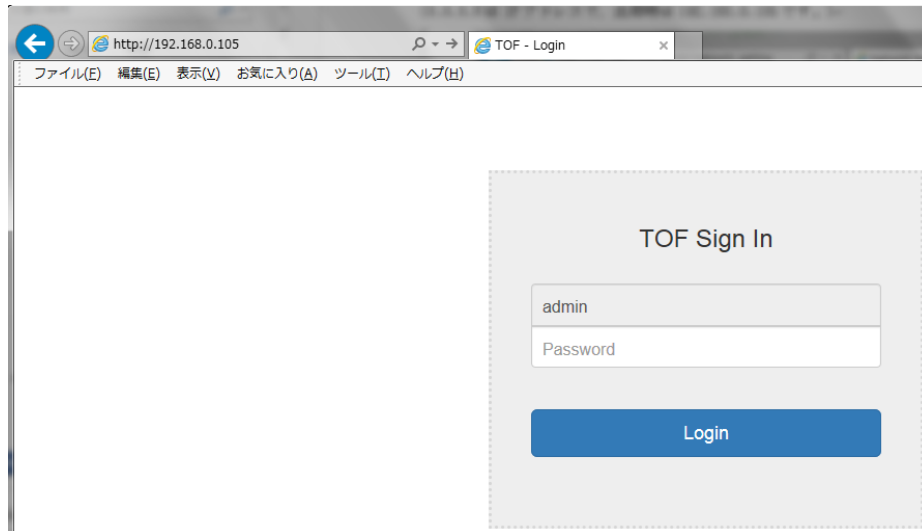
Figure 7 Network Setting

#### 4.5.2.2 For models Web GUI is available

Inputting the following URL to internet browser, and login on login page as Figure 8. Default password is admin. (User name and password must be changed.)

**http://X.X.X.X**

(X.X.X.X is IP address, and factory default is 192.168.0.105.)



**Figure 8 Login page of Web GUI**

After login, and select Network in the left menu, network page of Figure 8 is shown to set network.

## Network

IP Address	<input type="text" value="192.168.0.105"/>
Subnet Mask	<input type="text" value="255.255.255.0"/>
Default Gateway	<input type="text" value="192.168.0.1"/>
Default DNS	<input type="text" value="8.8.8.8"/>
Additional DNS	<input type="text"/>

**Figure 9 Network setting on Web GUI**

### 4.5.3 Switch Hardware Type

Hardware type (TOFv1/TOFv2) can be switched (Except of old firmware of HLS-LFOM1). Switching hardware type shall be done during application and SDK are stopped (It cannot be switched by SDK). The setting will be saved in TOF sensor.

#### 4.5.3.1 For models Web GUI is not available

The utility tool (tof\_utility.exe only or Windows) is used. Inputting IP address, selecting TOFv1/TOFv2 and push Set button(Refer to Figure 10).

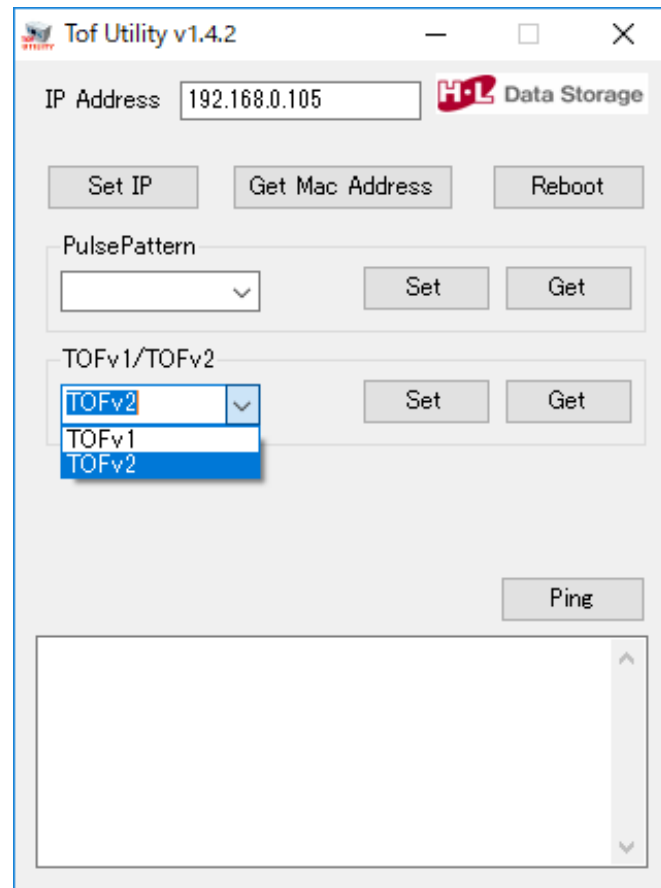
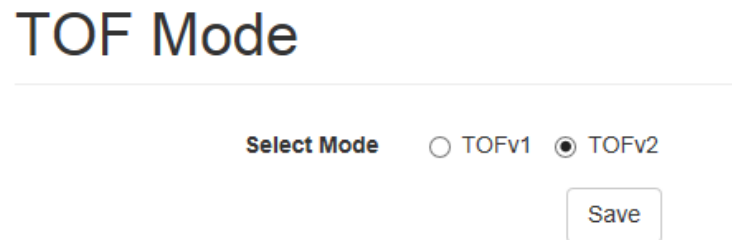


Figure 10 Switch hardware type (with tof\_utility.exe)

#### 4.5.3.2 For models Web GUI is available

Login Web GUI as well as setting IP address (Refer to #4.5.2.24.5.2.1), select TOF Mode in the left menu, select TOFv1/TOFv2, and push Save button on the page (Figure 11).



The screenshot shows a web interface titled "TOF Mode". Below the title is a horizontal line. Underneath the line, there is a label "Select Mode" followed by two radio button options: "TOFv1" and "TOFv2". The "TOFv2" option is selected, indicated by a filled circle. To the right of these options is a rectangular button labeled "Save".

**Figure 11 Switch hardware type (with Web GUI)**

#### 4.5.4 Interference Prevention

When multiple TOF sensors are used simultaneously in a place, interference causes worse of depth data accuracy. If interference occurs, it can be seen by eyes looking periodic fluctuations on depth image such as Tof2dViewer.exe (Refer to #7.1.1).

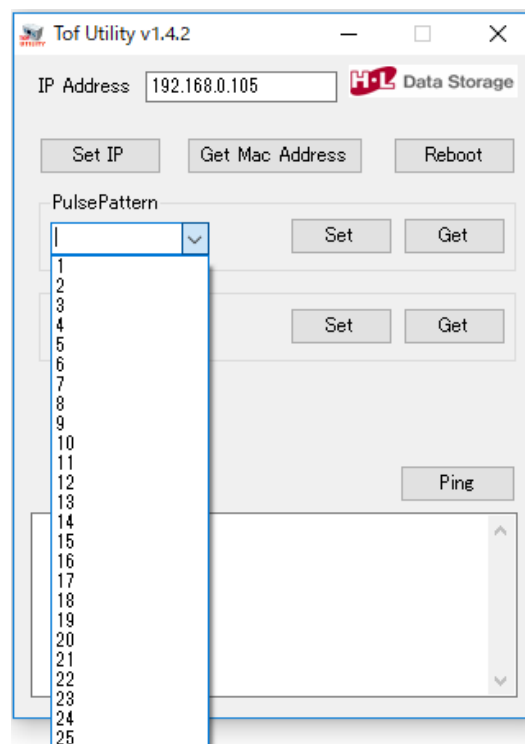
Setting a different pulse pattern to each TOF sensor, maximum 25 sensors (in case of 1.5x distance mode) can be used simultaneously in a place. As Table 4-13, it must be cared that available pulse pattern numbers are different depending on distance mode. If sensors which have different distance modes are used simultaneously, interference may occur despite different pulse patterns, and suitable number should be selected looking depth image. Setting of pulse pattern will be saved in TOF sensor, however it will be available after rebooting TOF sensor. **TOF sensor must be rebooted after changing pulse pattern.**

**Table 4-13 Interference Prevention Pattern**

	Pulse Pattern Number	
	Min Value	Max Value
0.5x mode(3.3m)	1	17
1.0x mode(6.6m)	1	11
1.5x mode(9.9m)	1	25
2.0x mode(13.2m)	1	16

##### 4.5.4.1 For models Web GUI is not available

The utility tool (tof\_utility.exe only or Windows) is used. Inputting IP address, selecting Pulse Pattern, and push Set button(Refer to Figure 12). After setting, The TOF sensor shall be rebooted.



**Figure 12 Interference Prevention (with tof\_utility.exe)**

#### 4.5.4.2 For models Web GUI is available

Login Web GUI as well as setting IP address (Refer to #4.5.2.24.5.2.1), select Interference Prevention in the left menu, select pulse pattern for distance mode which is used, push Save button on the page (Figure 13), and reboot the TOF sensor.

The screenshot shows a web interface titled "Interference Prevention". It features a table with two rows: "Distance Mode" and "Pulse Pattern". The "Distance Mode" row has four columns with values: "0.5x(3.3m)", "1.0x(6.6m)", "1.5x(9.9m)", and "2.0x(13.2m)". The "Pulse Pattern" row has four corresponding dropdown menus, each with a downward arrow icon. Below the table, there is a "Save" button.

Distance Mode	0.5x(3.3m)	1.0x(6.6m)	1.5x(9.9m)	2.0x(13.2m)
Pulse Pattern	▼	▼	▼	▼

Save

**Figure 13 Interference Prevention (with Web GUI)**

#### 4.5.5 Setting Upside Down (HLS-LFOM5)

Since image direction of HLS-LFOM5 is normal when sensor is put on a ceil (however it is rotated 90degrees), it becomes opposite when sensor is put on a wall(Figure 14).

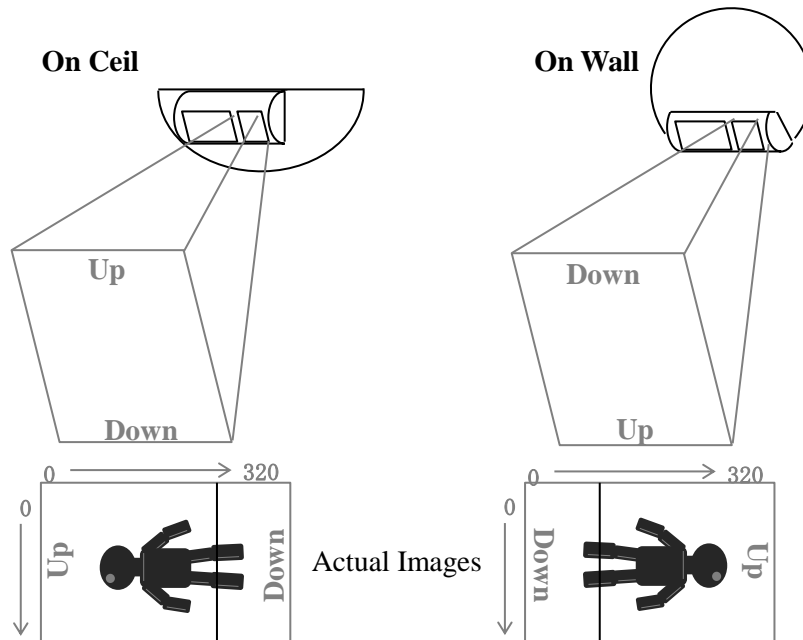


Figure 14 Image Direction of HLS-LFOM5

Login Web GUI as well as setting IP address (Refer to #4.5.2.24.5.2.1), select Image Direction in the left menu, and select Opposite if the sensor is put on a wall, then image direction becomes same as putting on a ceil (Figure 15). The setting will be saved in TOF sensor. It can be changed during SDK is working however it cannot be set from SDK.

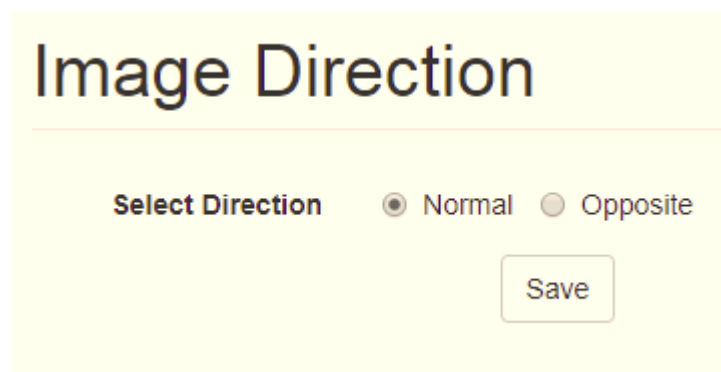


Figure 15 Setting Upside Down

#### **4.5.6 Time**

TBD



#### 4.5.7 Register to initialization file

To access the TOF sensor from SDK, it must be registered to initialization file explained in #5.12. Default initialization file is generated by `TofManager::Initialize()` method, and a new TOF sensor can be added by `TofManager::AddTof()` method. Otherwise, initialization file can be directory edited.

To register a TOF sensor to initialization file,

- (1) TOF ID
- (2) MAC address
- (3) IP address
- (4) RTP port number

are necessary. However, if (1)(2) and (4) except of IP address are empty, SDK sets such value. At least only IP address must be set. If (4) is set 0 or empty, it is set automatically by SDK. Even if (4) is set 0 or empty, 0 is set at initialization file. Only IP address should be set same as #4.5.2 setting.

An example of initialization file which only IP address(192.168.0.105) is registered is in /sample folder. If it is used, it should be copied to same folder with user application (.exe) or the folder which a member of `TofManager` indicates.

## 5 Software Configuration

### 5.1 System Configuration

Figure 21 shows the system configuration using the TOF sensor and this SDK. This SDK shall be installed in Windows PC or Server with user application, and can control multiple TOF sensors through a network.

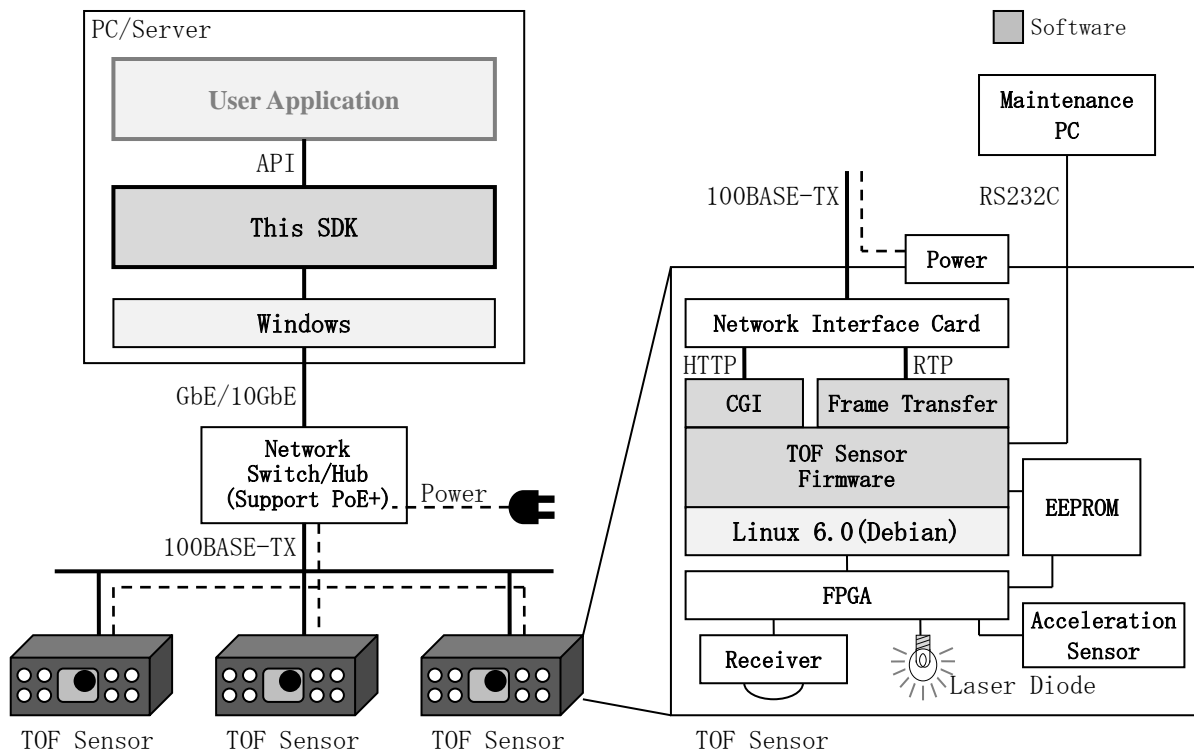


Figure 16 System Configuration

## 5.2 System Performance

Table 5-1 and Table 5-2 shows possible frame rate per a sensor depending on number of sensors and picture size. Please configure the system referring to Table 5-1 in case of 1G bit Ethernet (GbE) of PC/Server side, and Table 5-1 in case of 100M bit Ethernet (100BASE). The hatching items mean it cannot achieve to the maximum frame rate of 30fps. Since the network port of the TOF sensor is 100M bit Ethernet, frame rate cannot achieve to 30fps in case of default picture size (640 x 480).

Network transition efficiency is 80% in the calculation. Only network bottleneck is considered in this performance estimation, and other bottleneck such as PC/Server performance is not considered.

**Table 5-1 Frame Rate depending on number of sensors and picture size (GbE above Switch/Hub)**

Picture size	1/1 (640x480)	1/2 (320x240)	1/4 (160x120)	1/8 (80x60)	1/10 (64x48)	1/16 (40x30)	1/20 (32x24)
Amount(byte)	614,400	153,600	38,400	9,600	6,144	2,400	1,536
1 Sensor	7fps	30fps	30fps	30fps	30fps	30fps	30fps
2 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
3 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
4 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
5 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
6 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
7 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
8 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
9 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
10 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
11 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
12 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
13 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
14 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
15 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
16 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
17 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
18 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
19 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
20 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
21 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
22 Sensors	7fps	29fps	30fps	30fps	30fps	30fps	30fps

**Table 5-2 Frame Rate depending on number of sensors and picture size (100BASE above Switch/Hub)**

Picture size	1/1 (640x480)	1/2 (320x240)	1/4 (160x120)	1/8 (80x60)	1/10 (64x48)	1/16 (40x30)	1/20 (32x24)
Amount(byte)	614,400	153,600	38,400	9,600	6,144	2,400	1,536
1 Sensor	7fps	30fps	30fps	30fps	30fps	30fps	30fps
2 Sensors	7fps	30fps	30fps	30fps	30fps	30fps	30fps
3 Sensors	5fps	21fps	30fps	30fps	30fps	30fps	30fps
4 Sensors	4fps	16fps	30fps	30fps	30fps	30fps	30fps
5 Sensors	3fps	13fps	30fps	30fps	30fps	30fps	30fps
6 Sensors	2fps	10fps	30fps	30fps	30fps	30fps	30fps
7 Sensors	2fps	9fps	30fps	30fps	30fps	30fps	30fps
8 Sensors	2fps	8fps	30fps	30fps	30fps	30fps	30fps
9 Sensors	1fps	7fps	28fps	30fps	30fps	30fps	30fps
10 Sensors	1fps	6fps	26fps	30fps	30fps	30fps	30fps
11 Sensors	1fps	5fps	23fps	30fps	30fps	30fps	30fps
12 Sensors	1fps	5fps	21fps	30fps	30fps	30fps	30fps
13 Sensors	1fps	5fps	20fps	30fps	30fps	30fps	30fps
14 Sensors	1fps	4fps	18fps	30fps	30fps	30fps	30fps
15 Sensors	1fps	4fps	17fps	30fps	30fps	30fps	30fps
16 Sensors	1fps	4fps	16fps	30fps	30fps	30fps	30fps
17 Sensors	0fps	3fps	15fps	30fps	30fps	30fps	30fps
18 Sensors	0fps	3fps	14fps	30fps	30fps	30fps	30fps
19 Sensors	0fps	3fps	13fps	30fps	30fps	30fps	30fps
20 Sensors	0fps	3fps	13fps	30fps	30fps	30fps	30fps
21 Sensors	0fps	3fps	12fps	30fps	30fps	30fps	30fps
22 Sensors	0fps	3fps	11fps	30fps	30fps	30fps	30fps

### 5.3 Coordinate System

This software uses 2 types of coordinate system (Refer to Figure 22).

The Camera Coordinate system (left of Figure 22) consists of depth data (distance between objects) in each pixel captured by camera. Since it is narrowed down by lens of camera, if a flat wall is captured just in front for example, the depth data around the center of the picture ( $z_0$  of Figure 22) and the depth data around edge of the picture ( $z_1$  in left of Figure 22) are different.

The other hand, if the camera coordinate data is converted into 3D, each point corresponding to each pixel is converted to 3D coordinate in the 3D Coordinate system (right of Figure 22). The coordinate system uses right-handed system. The position of the receiving part of the sensor is  $(x,y,z) = (0,0,0)$ , and the direction of the sensor facing is plus of z-coordinate in the 3D Coordinate system. If a flat wall is captured just in front for example, z-coordinates of all points on the flat wall are same ( $z_0$  in right of Figure 22).

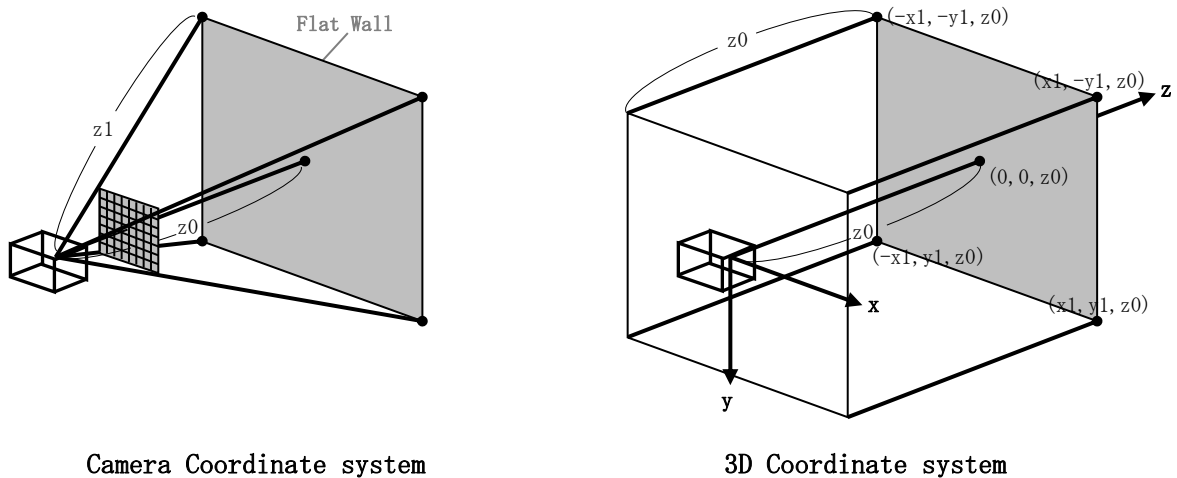


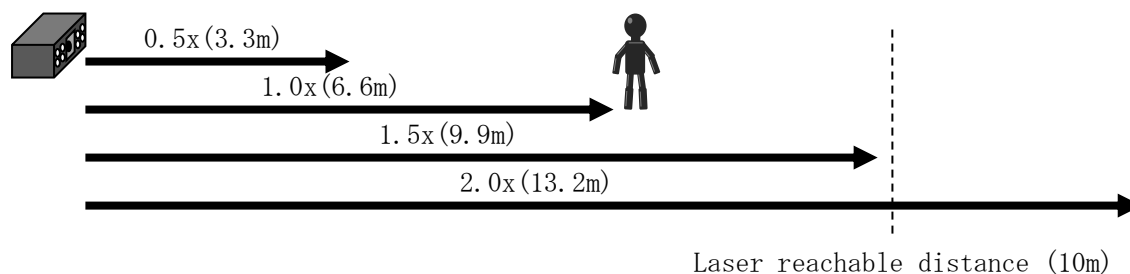
Figure 17 Coordinate System

## 5.4 Detection Distance

### 5.4.1 Distance Mode

This TOF sensor supports distance modes as Figure 23 and Table 5-3. Distance mode is set as magnification as 1.0x is 6.6m which is default maximum measurable distance. Since max laser reachable distance is 10m, over 10m is out of specification despite 2.0x mode.

Please refer to chapter 3. Hardware specification about detect distance.



**Figure 18 Distance Mode**

**Table 5-3 Distance Mode**

Distance Mode	DistanceMode	Max Distance	Remark
0.5x	dm_0_5x	3.3m	
1.0x	dm_1_0x	6.6m	Default
1.5x	dm_1_5x	9.9m	
2.0x	dm_2_0x	13.2m	but over 10m is out of spec.

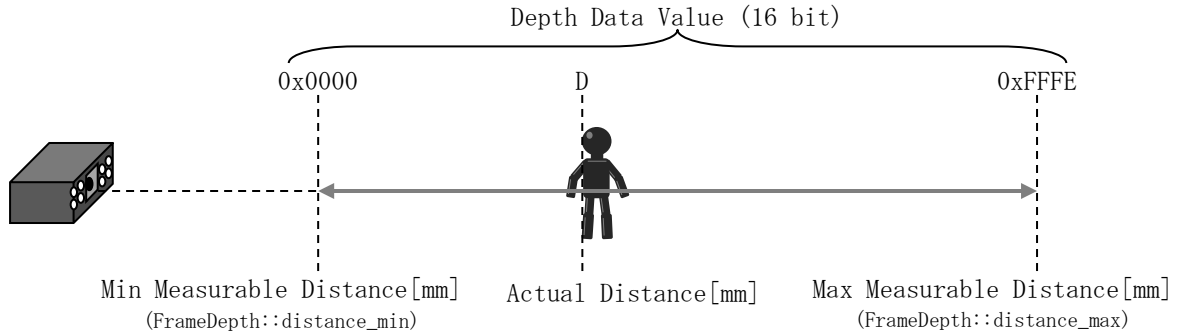
Regardless of distance mode, depth data is measured as 0 to 0xFFFFE(TOFv1) or 0 to 0xFE(TOFv2), resolution of delta 1 is depending on depth mode as Table 5-4.

**Table 5-4 Distance Accuracy**

Distance Mode	Max Distance	Resolution(Distance of $\Delta 1$ )		Remark
		TOFv1 (2byte/pixel)	TOFv2 (1byte/pixel)	
0.5x	3.3m	0.05mm	12.8mm	
1.0x	6.6m	0.10mm	25.6mm	
1.5x	9.9m	0.15mm	38.4mm	
2.0x	13.2m	0.20mm	51.2mm	

## 5.5 Convert to Distance

Depth data that is read to FrameDepth instance by `Tof::ReadFrame()` method of `Tof` class is 16 bit per a pixel. This value is, 0x0000 at minimum measurable distance, 0xFFFE at maximum measurable distance (Refer to Figure 24).



**Figure 19 Convert to Distance**

If a depth data of a pixel is  $D$ , the actual distance can be calculated as the following formula.

$$\text{Actual Distance} = \frac{\text{Max Measurable Distance} - \text{Min Measurable Distance}}{0xFFFE} \cdot D + \text{Min Measurable Distance}$$

When depth data is read to FrameDepth instance by `Tof::ReadFrame` method of `Tof` class, minimum measurable distance is set in `distance_min` member of FrameDepth instance, and maximum measurable distance is set in `distance_max` member of FrameDepth instance. when actual distance is calculated, such members should be used.

Maximum measurable distance can be set distance mode (`Tof::SetDistanceMode()`).

## 5.6 Coodinate Rotation

3D coordinate can be rotated by `Frame3d::Rotate()`/`Frame3d::RotateZYX()` method. Rotation order and rotation direction are shown in Figure 25. `Frame3d::Rotate()` method rotates in order of x-axis, y-axis, and z-axis, `Frame3d::RotateZYX()` method rotates in order of z-axis, y-axis, and x-axis. The rotation direction is clockwise toward the positive direction of each axis. Sensor installation angle set in `Tof::SetAttribute()` is also same.

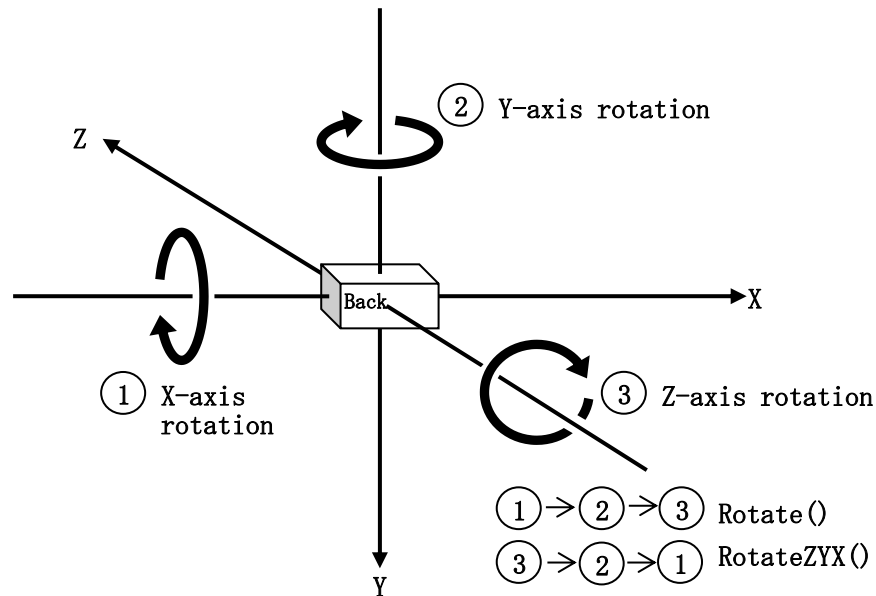


Figure 20 3D Coordinate Rotation

To convert 3D data gotten by TOF sensor to top down view, the TOF sensor facing direction (Z-axis) becomes directly below after rotating 3D data by the TOF sensor angle. (Refer to Figure 26)

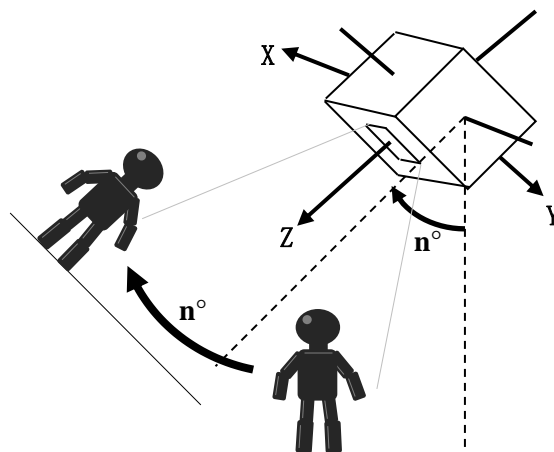
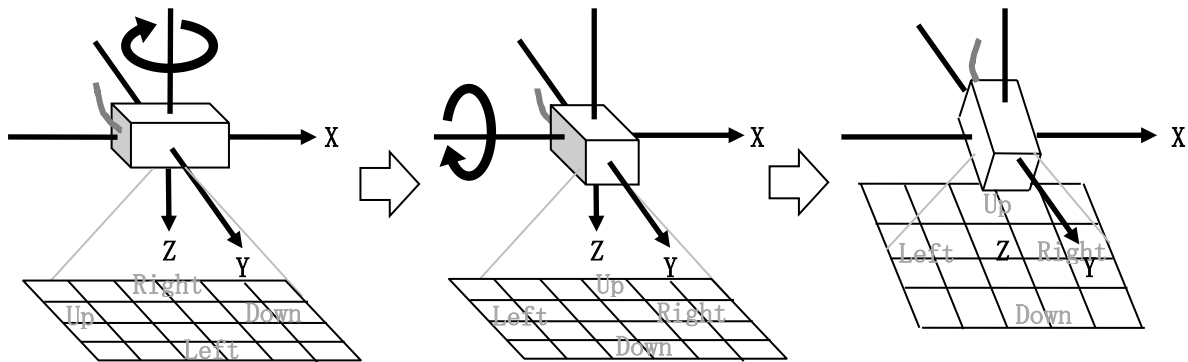


Figure 21 Convert to top down view



When like as a narrow aisle is captured, the sensor should be rotated 90 degrees so that the wider FOV becomes vertical and it can cover the area effectively. In the case, rotating 90 degrees around z-axis then rotating around x-axis to adjust angle of the sensor. Frame3d::RotateZYZ() method is used for the case to rotate z-axis, y-axis, x-axis order.



**Figure 22 Rotating sensor 90 degrees**

## 5.7 Image Size

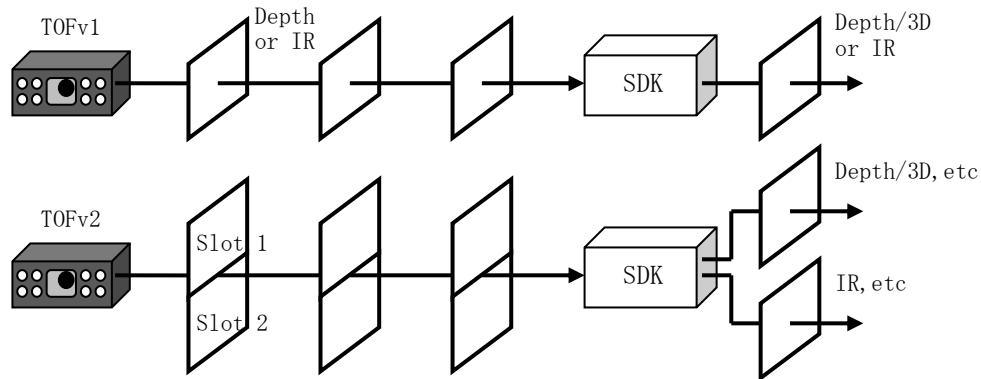
This TOF sensor supports image sizes as Table 5-5. TOFv2 sensor does not support VGA (640 x 480). Data sent from TOFv2 sensor is 1 byte / pixel against to 2 bytes / pixel of TOFv1, however this SDK provides 2 bytes / pixel despite TOFv2 sensor.

**Table 5-5 Image Size**

Scale	Number of pixels			Image Size		Transfer Rate(in case of 30fps)		Remark
	W	H	Total	TOFv1 (2byte/pixel)	TOFv2 (1byte/pixel)	TOFv1 (2byte/ pixel)	TOFv2 (1byte/ pixel)	
1/1	640	480	307,200	614,400byte	N/A	20,275.2KB/s (162.2Mbps)	N/A	VGA
1/2	320	240	76,800	153,600byte	76,800byte	5,068.8KB/s (40.6Mbps)	2,534.4KB/s (20.3Mbps)	QVGA
1/4	160	120	19,200	38,400byte	19,200byte	1,267.2KB/s (10.1Mbps))	633.6KB/s (5.1Mbps)	QQVGA
1/8	80	60	4,800	9,600byte	4,800byte	316.8KB/s (2.5Mbps)	158.4KB/s (1.3Mbps)	
1/10	64	48	3,072	6,144byte	3,072byte	202.8KB/s (1.6Mbps)	101.4KB/s (0.8Mbps)	
1/16	40	30	1,200	2,400byte	1,200byte	79.2KB/s (0.6Mbps)	39.6KB/s (0.3Mbps)	
1/20	32	24	768	1,536byte	768byte	50.7KB/s (0.4Mbps)	25.3KB/s (0.2Mbps)	

## 5.8 Frame Data

Frame data outputted from TOFv1 sensor is either of depth data or IR data, however TOFv2 sensor can output 2 types of frame data in slot 1 and slot 2 simultaneously (Refer to Figure 28). Thus depth data and IR image at a same moment can be captured by TOFv2 sensor.



**Figure 23 Frame Data**

Depth data or IR image can be alternatively selected in case of TOFv1 sensor, however TOFv2 sensor can send depth data, IR image, motion data and background data of background subtraction on slot 1 and slot 2 simultaneously as Table 5-6. Frame data can be set through `Tof::SetCameraMode()`.

**Table 5-6 Kinds of Frame Data**

CameraMode	TOFv1	TOFv2		Remark
		Slot 1	Slot 2	
CameraModeDepth	Depth	Depth	None	
CameraModeIr	IR	IR	None	
CameraModeMotion	N/A	Motion	None	
CameraModeBackground	N/A	Background	None	
Depth_Motion	N/A	Motion + Background	Motion	
Depth_Background	N/A	Motion + Background	Background	
Depth_Ir	N/A	Motion + Background	IR	
Motion_Background	N/A	Motion	Background	
Motion_Ir	N/A	Motion	IR	
Background_Ir	N/A	Background	IR	

To get 2 data from SDK which sent from TOFv2 sensor on slot 1 and slot 2 together, `Tof::ReadFrame()` with 2 arguments is used.

## 5.9 Class

The class configuration is shown on Figure 29 which is opened by the API.

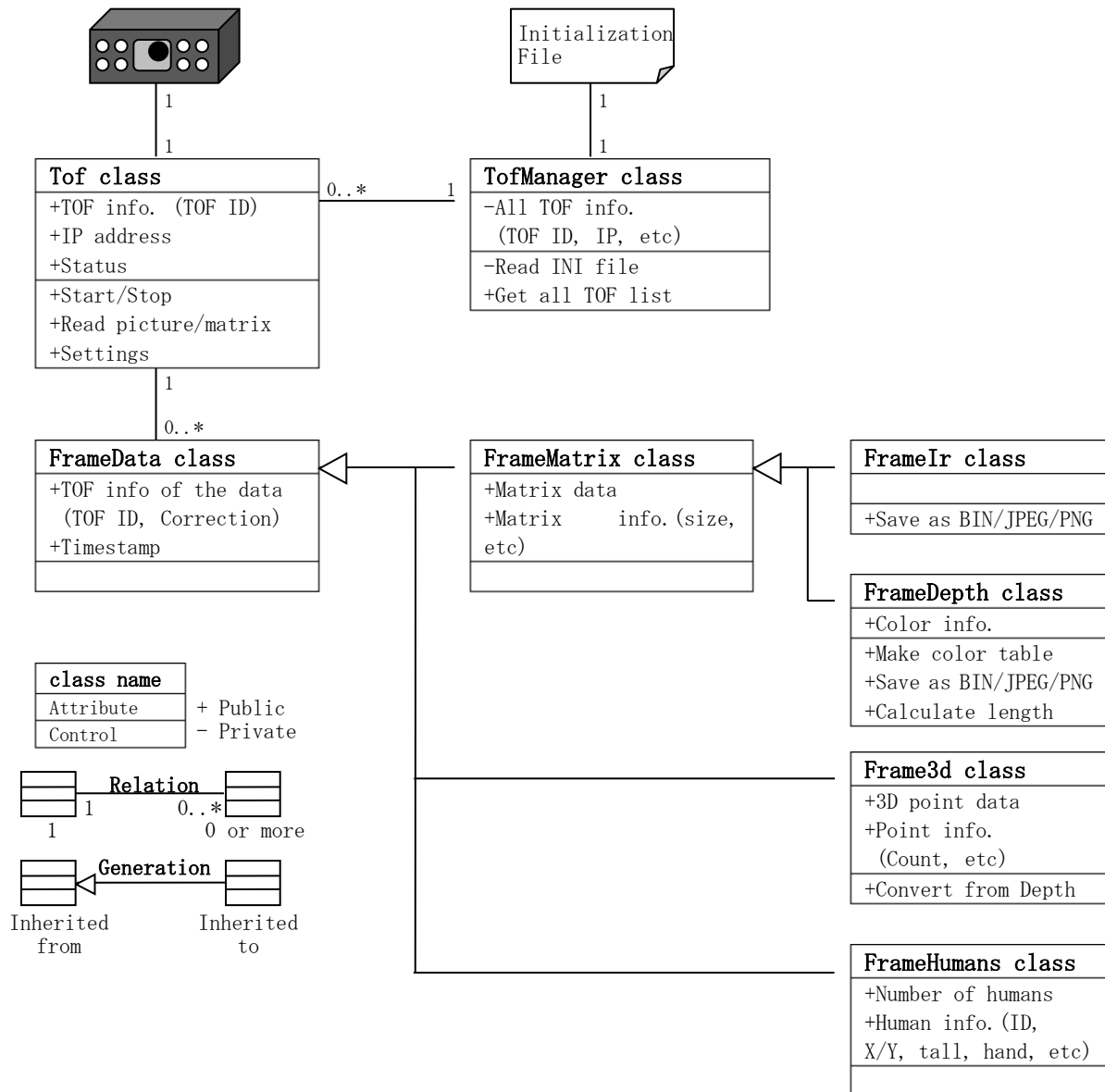


Figure 24 Class Diagram

## 5.10 State Transition Diagram

Figure 30 shows the state transition diagram of Tof instance. Please create Tof instance for each TOF sensor. Please create instances for all TOF sensors controlled by user application. TOF sensor cannot be controlled until a specific TOF sensor is assigned to the instance by Tof::Open() method even after created. A specific TOF sensor is assigned inputting the TOF information of the TOF sensor through augment of Tof::Open() method. TOF information of all TOF sensors managed by the SDK can be gotten by TofManager::GetTofList() method of TofManager class.

The TOF sensor starts measurement and data transfer by Tof::Run() method. Measured data can be read to FrameDepth (or FrameIr, Frame3d) instance by Tof::ReadFrame() method during running.

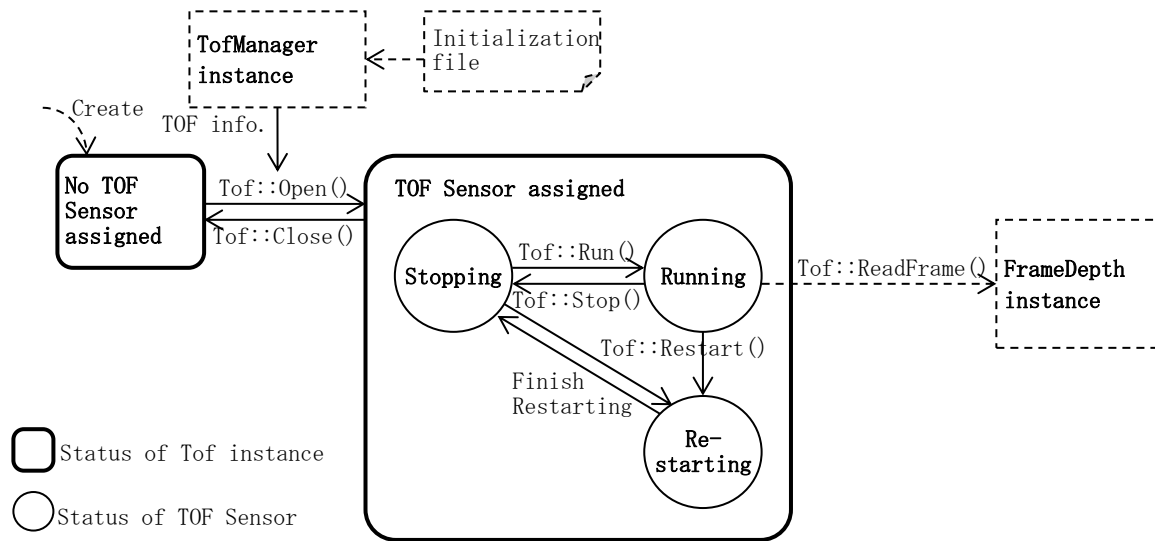


Figure 25 State transition diagram of Tof instance

## 5.11 Sequence

### 5.11.1 Start Sequence

The start sequence of TOF sensor is shown in Figure 31. At first, creating TofManager instance, reading initialization file (Refer to chapter 5.12) by TofManager::Open() method, getting the number of TOF sensors and a list of TOF information of all TOF sensors managed by the SDK by TofManager::GetTofList() method, and creating Tof instances for all TOF sensors. Tof::Open() method of Tof instances shall be called after the target TOF sensor is powered on. Settings for each TOF sensor shall be before starting transferring by Tof::Run() method.

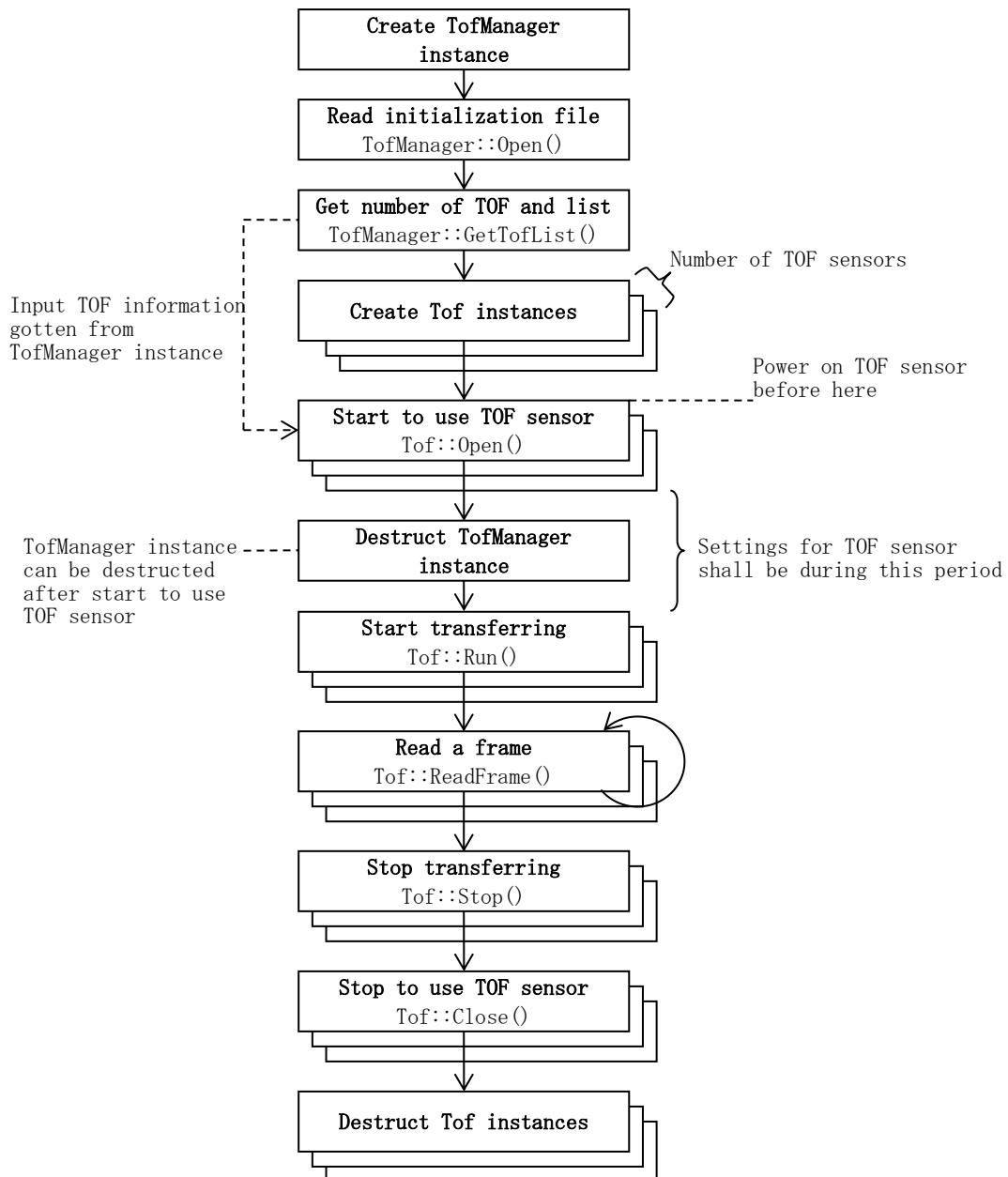


Figure 26 TOF sensor start sequence

### 5.11.2 Read Frame Sequence

There 2 methods of synchronous and asynchronous to read a frame by `Tof::ReadFrame()` method of `Tof` instance.

#### 5.11.2.1 Read Frame Sequence (Synchronous)

Figure 32 shows Read Frame sequence of synchronous method. If `Tof::ReadFrame()` method is called between starting transfer by `Tof::Run()` method and the first Frame arrival, it waits until the first Frame arrival in `Tof::ReadFrame()` method (the case of Frame 0 of Figure 32).

Just after the latest Frame is read to `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance by `Tof::ReadFrame()` method, if `Tof::ReadFrame()` method is called again inputting same `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance, `Tof::ReadFrame()` method refers the timestamp of inputted `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance and waits for a new Frame arrival to avoid reading same Frame again (the cases of Frame 1,2,4 of Figure 32). In this case, response of `Tof::ReadFrame()` method may be delayed maximum 1 Frame time (33ms).

If a new Frame already arrived when `Tof::ReadFrame()` method is called, `Tof::ReadFrame()` method immediately reads Frame data into inputted `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance without wait, and returns (the cases of Frame 3,6 of Figure 32).

If a next call of `Tof::ReadFrame()` method is delayed after previous call, a Frame lost does not occur in case of shorter than 2 Frames time (66ms) (the case of Frame 3 of Figure 32), however it occurs in case of longer than 2 Frames time (66ms) (the case of Frame 6 of Figure 32).

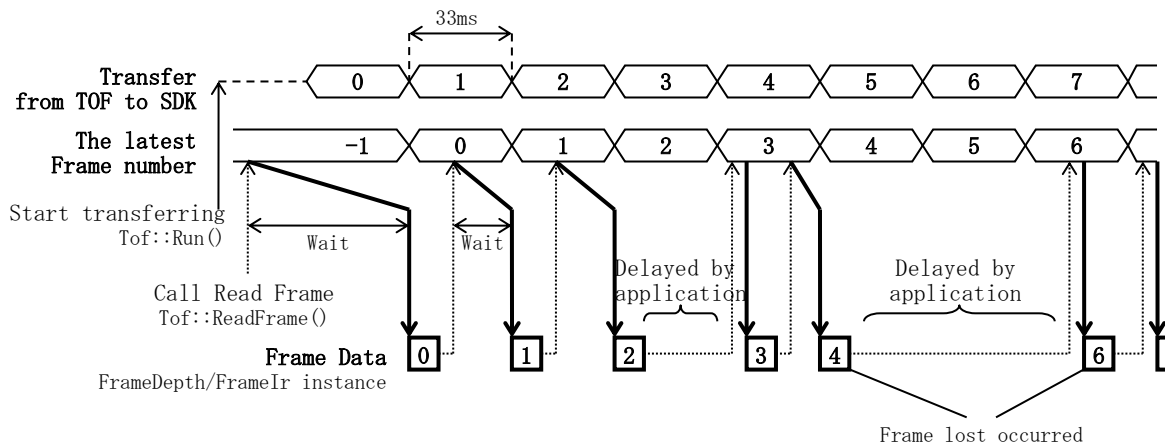
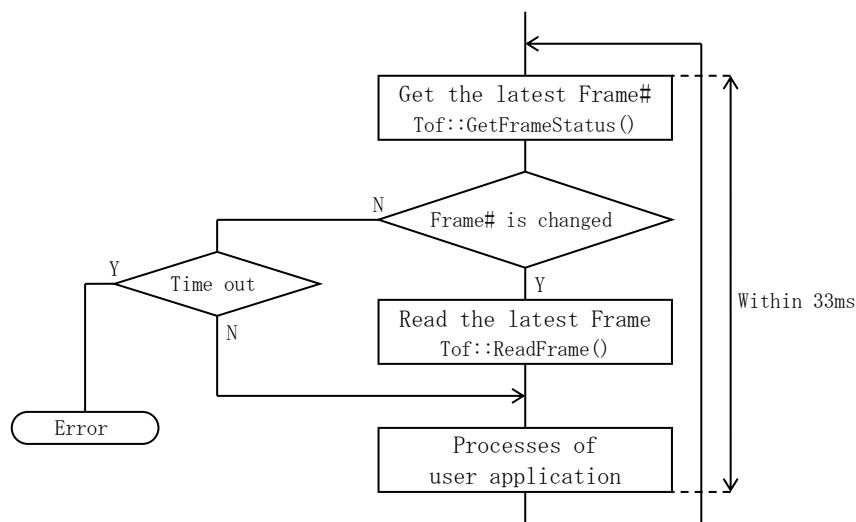


Figure 27 Read Frame Sequence (Synchronous)

### 5.11.2.2 Read Frame Sequence (Asynchronous)

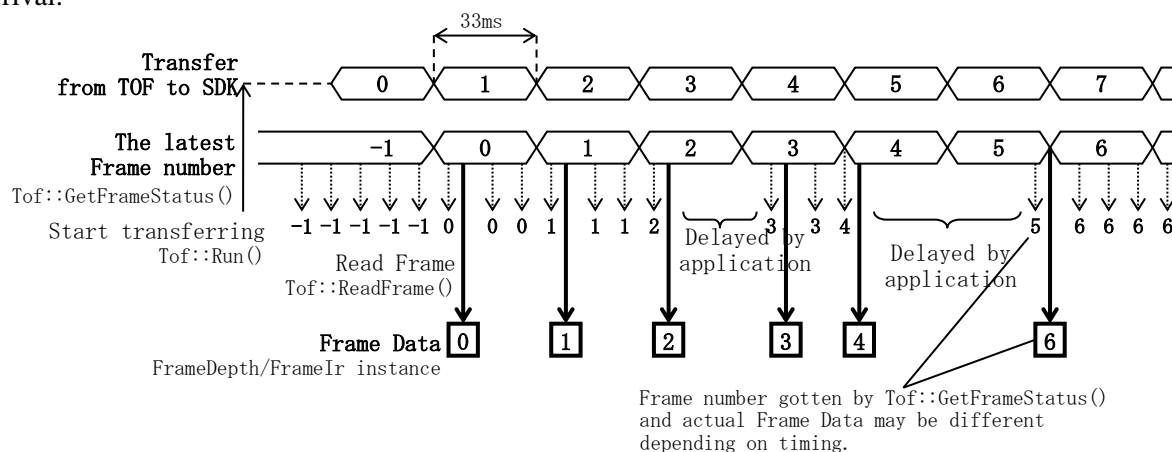
Since waiting for a next Frame may occur in synchronous method, response of `Tof::ReadFrame()` method may be delayed and user application cannot work during the period. To avoid the wait in `Tof::ReadFrame()` method, `Tof::ReadFrame()` method should be called after confirming a next new Frame arrived by `Tof::GetFrameStatus()` method.

The flowchart of Read Frame sequence using `Tof::GetFrameStatus()` method is shown in Figure 33. Since `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance storing Frame data read by `Tof::ReadFrame()` method has Frame number and timestamp, after confirming the latest Frame number or timestamp is updated through `Tof::GetFrameStatus()` method from previous Frame number or timestamp in `FrameDepth` (or `FrameIr`, `Frame3d`, etc) instance, `Tof::ReadFrame()` method should be called to read a new Frame.



**Figure 28 Flowchart of Read Frame Sequence (Asynchronous)**

Figure 34 shows Read Frame sequence of asynchronous method. `Tof::GetFrameStatus()` method returns Frame number as -1 between starting transferring by `Tof::Run()` method and the first Frame arrival.



**Figure 29 Read Frame sequence (Asynchronous)**



## 5.12 Initialization File

TofManager class manages all TOF sensors with Initialization File. Initialization File is written as XML format, and the format is shown in Table 5-7 and Figure 35. Initialization File is loaded by TofManager::Open() method of TofManager class. The folder location and the name of Initialization File are same folder of user application and "tof.ini" in default, however those can be changed in member variables of TofManager class. A new Initialization File is generated by TofManager::Initialize() method. Element values in Initialization File can be modified by manual however character code shall be UTF-8 when it is saved. Since Initialization File is important, it is recommended to make backup.

**Table 5-7 Initialization File**

Element			Contents
Level 1	Level 2	Level 3	
xml version=			XML version
Hlds			Root
	FileId		INI File identification
	IniVersion		INI File version
	Tof (x Number of Tof sensors)		TOF sensor information (for 1 sensor)
		Id	TOF ID(17 characters including ':')
		Mac	MAC address (17 characters including ':')
		Ip	IP address IPv4: Max 16 characters including '.' IPv6: Max 39 characters including ':' (* IPv6 is not supported yet)
		Rtp	RTP port number (If RTP port number is 0 or empty, it is set as random even number (from 49152 to 65535) not to overlap between TOFs.)

```
<?xml version="1.0" encoding="UTF-8"?>
<Hlds>
  <FileId>TofSdkIniFile</FileId>
  <IniVersion>1.0.0</IniVersion>
  <Tof>
    <Id>XX:XX:XX:XX:XX:XX</Id>
    <Mac>XX:XX:XX:XX:XX:XX</Mac>
    <Ip>XXX.XXX.XXX.XXX</Ip>
    <Rtp>XXXX</Rtp>
  </Tof>
  <Tof>
    <Id>XX:XX:XX:XX:XX:XX</Id>
    <Mac>XX:XX:XX:XX:XX:XX</Mac>
    <Ip>XXX.XXX.XXX.XXX</Ip>
    <Rtp>XXXX</Rtp>
  </Tof>
</Hlds>
```

**Figure 30 Example of Initialization File**

## 5.13 Features

### 5.13.1 Human Detection

TOF	TOFv1 or later
SDK	v1.3.0 or later

#### 5.13.1.1 Outline

Humans can be detected in depth data gotten by TOF sensor, and coordinate of each human can be outputted by this SDK. Since human coordinates are in a top down coordinate system regardless of TOF sensor position and installed angle, TOF sensor's installation is very flexible. The top down coordinate system is, X,Y coordinate of TOF sensor is (0,0), and right direction is positive direction of X-axis, and back direction is positive direction of Y-axis. (Refer to Figure 36)

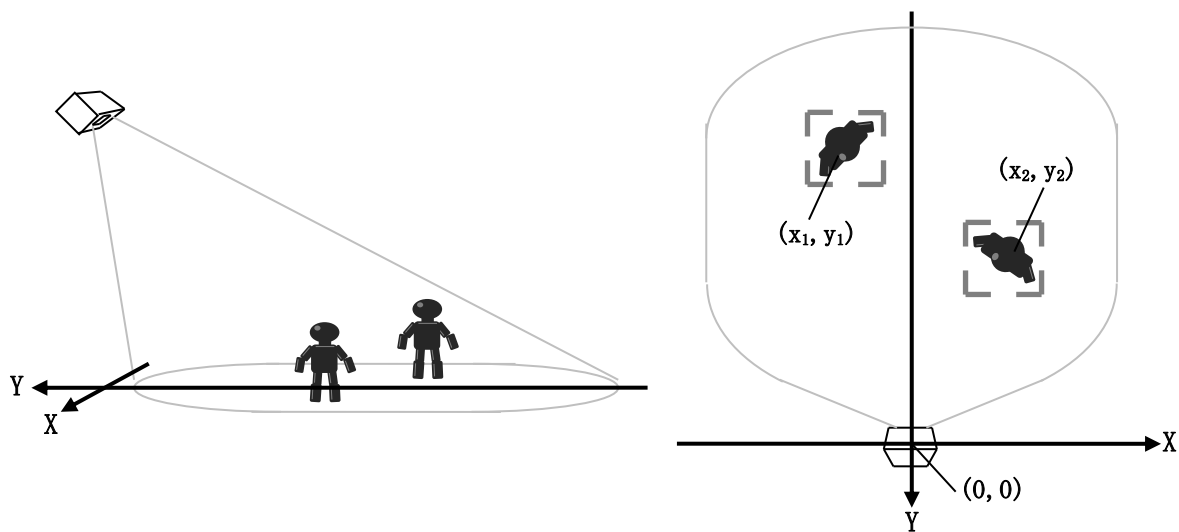


Figure 31 Human Detection feature

#### 5.13.1.2 Initial Settings

Since environment of detection area is automatically recognized, this feature can be started without fine tuning nor calibration. However to detect height from floor level and angle, only height (z) of TOF sensor from floor level, and sensor's angle (rx,ry,rz) shall be set through `Tof::SetAttribute()` method. The height shall be negative value as floor level is 0. (Refer to Figure 37)

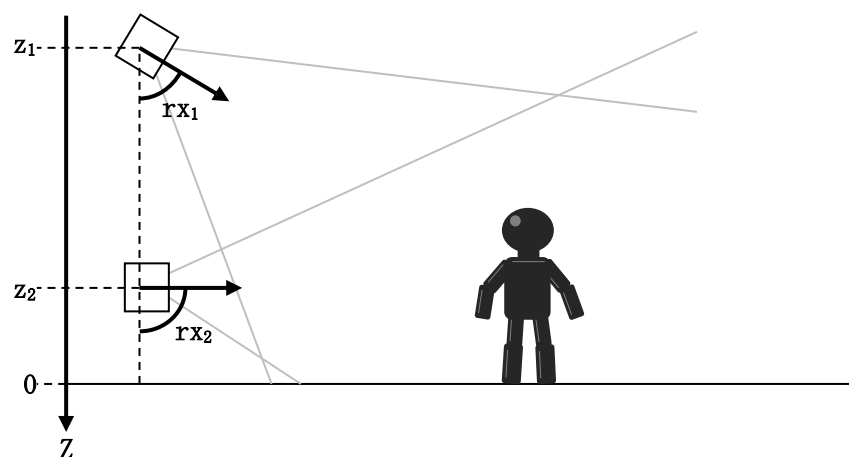
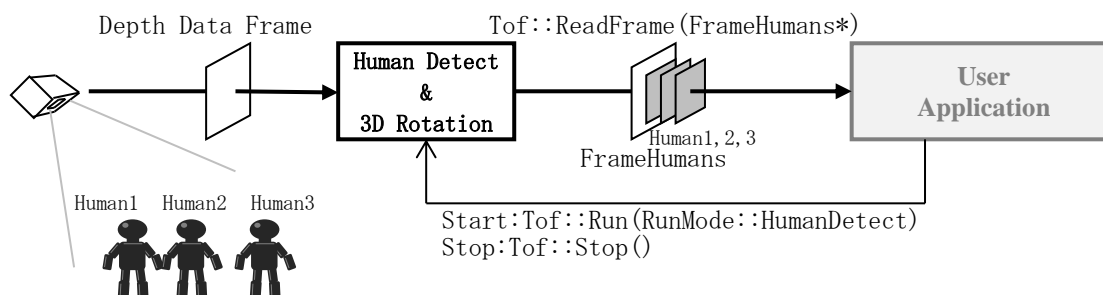


Figure 32 Initial Settings for Human Detection

### 5.13.1.3 Usage

Calling `Tof::Run()` method of `Tof` class with `RunMode::HumanDetect` as an argument, `Tof` instance starts receiving depth data from TOF sensor and human detection. After that, `FrameHuman` instance which includes all humans information detected in a frame can be read by `Tof::ReadFrame()` as well as reading other types of frame (Refer to #5.11.2 "Read Frame Sequence"). (Refer to Figure 38)



**Figure 33 Reading Human Frame**

### 5.13.1.4 Specification

Table 5-8 shows principal specification of Human Detection feature.

**Table 5-8 Specification of Human Detection feature**

Item	Specification	Remark
Number of detectable humans simultaneously	No limit in software	Depends on limitation of memory, however number of humans can exist in detected area is the first limitation.
Max detectable distance (Distance from sensor)	1.0x distance mode : 6.0m 1.5x distance mode : 7.0m	Even if longer distance is set, farther distance than this value cannot be detected.
Memory usage (Per a Tof instance)	100MB	
Max frame rate	30fps	
Supporting camera pixel	QVGA(320×240)	

### 5.13.1.5 Notes

- 1) Nobody should be in detected area when `Tof::Run()` is started.  
(If human is existing in the beginning, humans may not be detected a while.)
- 2) Human is not detected during the first 3 seconds after `Tof::Run()` started.
- 3) Other of QVGA mode is not supported. QVGA shall be set by `Tof::SetCameraPixel()`.

### 5.13.2 Capture/Emulation

TOF	TOFv1 or later
SDK	v1.3.0 or later

#### 5.13.2.1 Outline

Depth data received by ToF instance is written to a capture file. Starting ToF instance in emulation mode, depth data is read from a capture file without TOF sensor, and motions at capturing can be reproduced any number of times. (Refer to Figure 39)

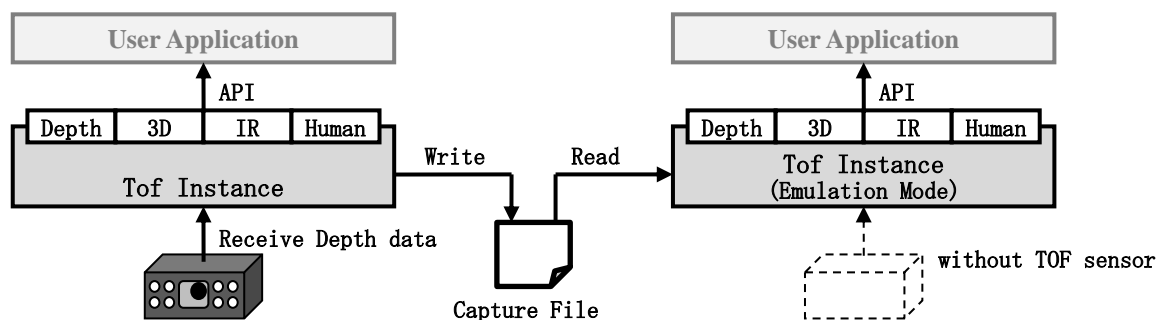


Figure 34 Capture/Emulation

#### 5.13.2.2 Capture Method

Table 5-9 and Figure 40 show capture method.

Table 5-9 Capture Operation

Operation	Description	Remark
ToF::CreateCaptureFile()	Set file name of capture file to create capture file.	If same file name is already existed, overwrite to the file.
ToF::capturetime	Set maximum capture duration. Default is 600 seconds (10 minutes)	If max capture duration is past, automatically stopped to avoid consuming storage due to forgetting stop.
ToF::Capture()	Set enable or disable capturing. Capturing is worked only if capture enabled and Run status.	
ToF::GetCaptureStatus()	Get capture status.	

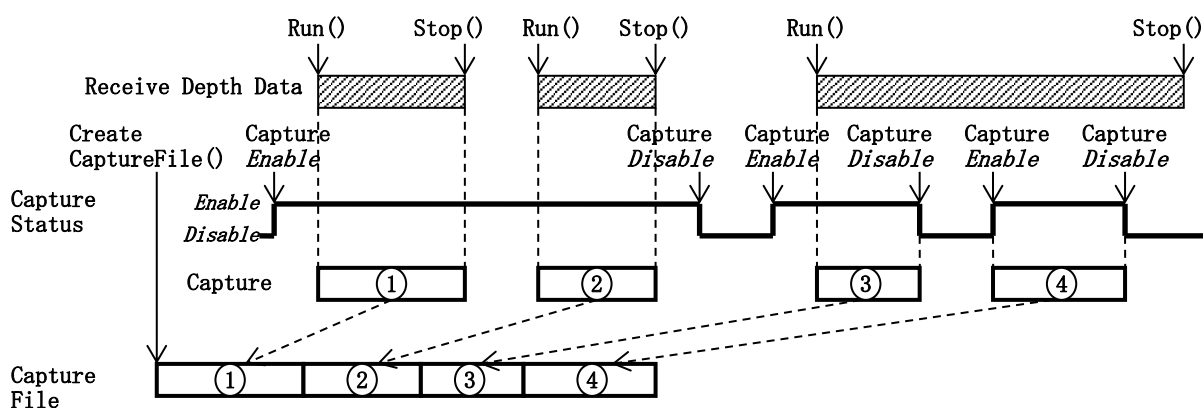


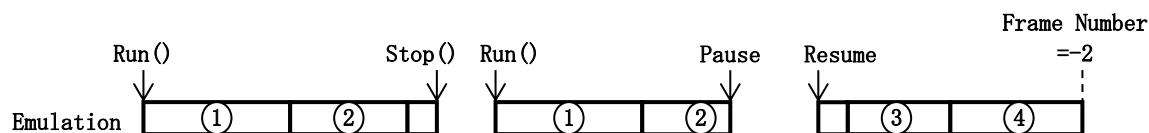
Figure 35 Capture Flow

### 5.13.2.3 Emulation Method

Table 5-10 and Figure 41 show emulation method.

**Table 5-10 Emulation Operation**

Operation	Description	Remark
Tof::Open()	Tof instance becomes emulation mode inputting a capture file. If there is no <Tof> tag in initialization file, emulation is possible without TOF sensor.	Settings of TOF sensor depends on the capture file.
Tof::Run()	Start to replay capture file. In emulation mode which TOF sensor is not used, frame by frame play is possible by Tof::ReadFrame() if FrameEmulation is set.	Replayed from the beginning of the capture file every Run() is called.
Tof::GetFrameStatus()	Frame number and timestamp at capturing is returned. Frame number is -2 at the end of capture file.	Frame data is all 0xFFFF at the end of capture file.
Tof::PauseEmulationTof()	Frame play can be paused if it is in emulation mode which TOF sensor is not used.	



**Figure 36 Emulation Flow**

### 5.13.2.4 Specification

Table 5-11 shows principal specification of capture feature.

**Table 5-11 Specification of capture feature**

Item	Specification	Remark
Capacity of capture file	4.6MB/second	In case of QVGA 30fps
Capturable data	Depth data IR data	Depth data and IR image can be captured simultaneously by TOFv2.

### 5.13.2.5 Notes

- 1) Don't change TOF sensor settings during capturing.  
(If TOF sensor setting is changed, restart from Tof::CreateCaptureFile().)
- 2) If start and stop capturing are done several times in a capture file, it is continuously replayed however timestamp is jumped.

### 5.13.3 Background Subtraction

TOF	TOFv2 or later
SDK	v2.0.0 or later

#### 5.13.3.1 Outline

Tofv2 sensor supports background subtraction that transfer background and motion object separately. This function remembers background at first, and treats objects as motion object which depth data is different with background (Refer to Figure 42). Since motion objects and background are separated by depth difference, motion object can be correctly detected if it is in front of background.

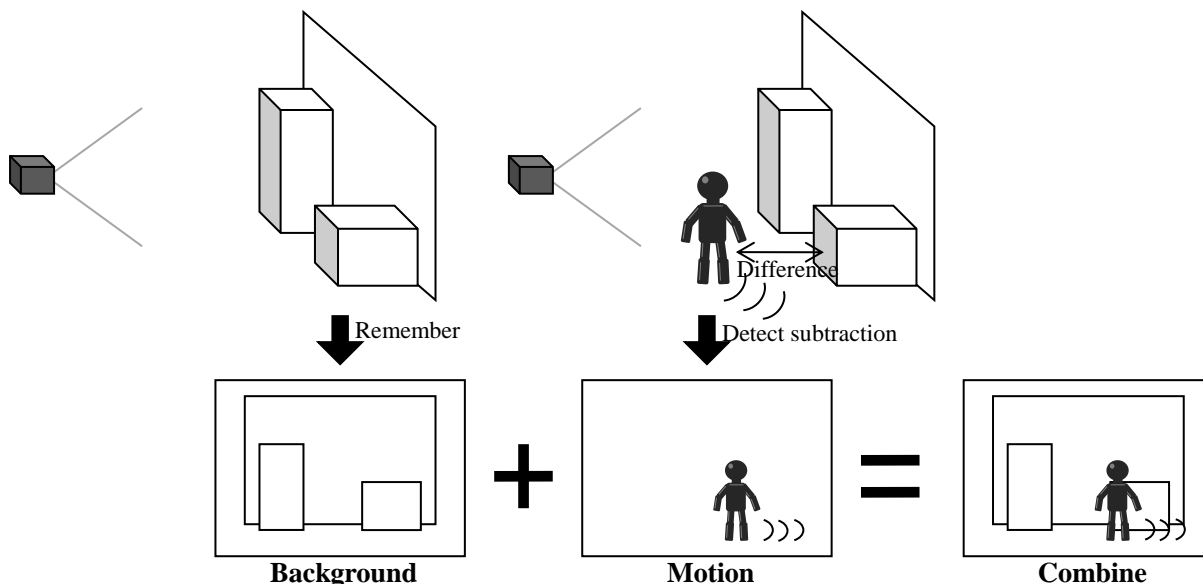


Figure 37 Background Subtraction

Since motion objects are detected in sensor, background area which is no change can be omitted to process in application. If motion frame is compressed in TOF sensor, data amount from TOF sensor can be reduced cutting data except of motion. However compression is not supported in v2.0.0 so far.

#### 5.13.3.2 Update Background

Background is updated as moving closer to motion object in specific quantity every update timing as Figure 43. Therefore, an object stays long period, it is swamped by background eventually.

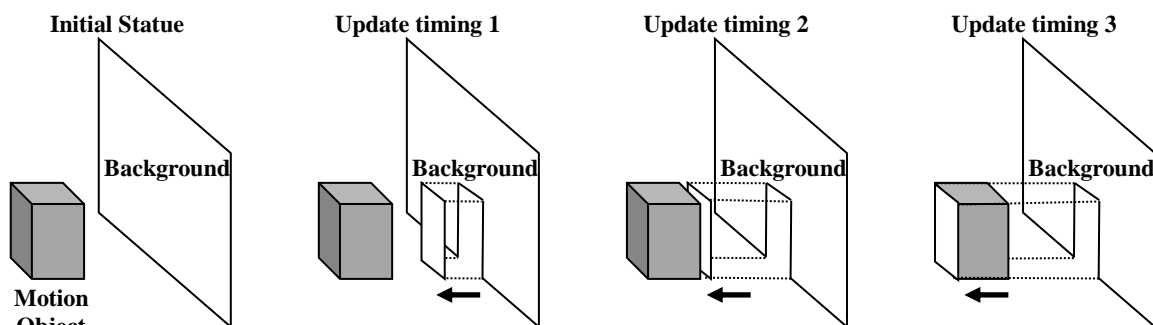


Figure 38 Update Background

### 5.13.3.3 Usage

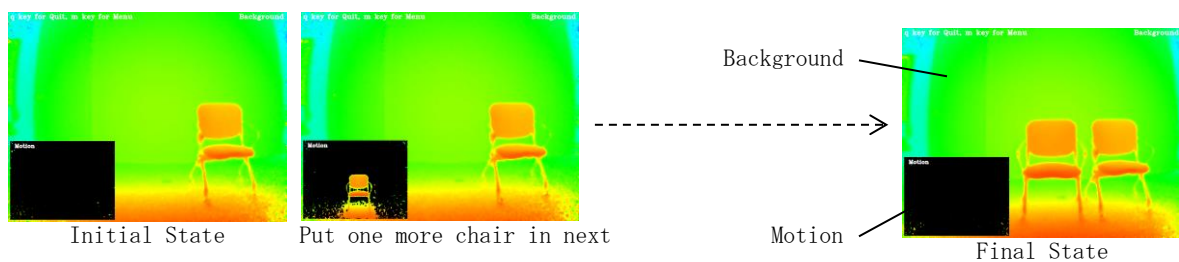
Update timing and update quantity are set by methods in Table 5-12. If background subtraction is not used, select depth frame through Tof::SetCameraMode() (Refer to #5.8).

**Table 5-12 Setting and Operation of Background Subtraction**

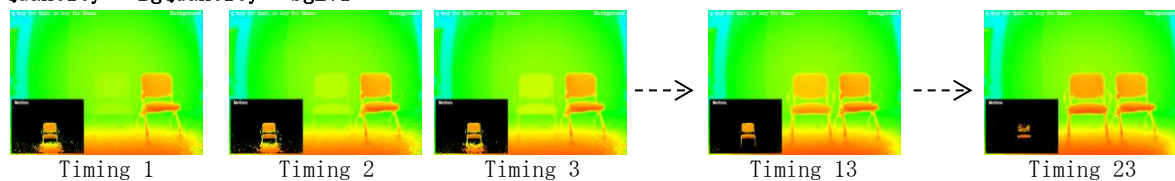
Setting/Operation	Discription	Remark
Tof::SetBackgroundInterval() Tof::GetBackgroundInterval()	Set or get interval of updating background	
Tof::SetBackgroundQuantity() Tof::GetBackgroundQuantity()	Set or get quantity of updating background	If background is not updated, select no update.
Tof::ResetBackground()	Uppdate background any time	

### 5.13.3.4 Image Samples

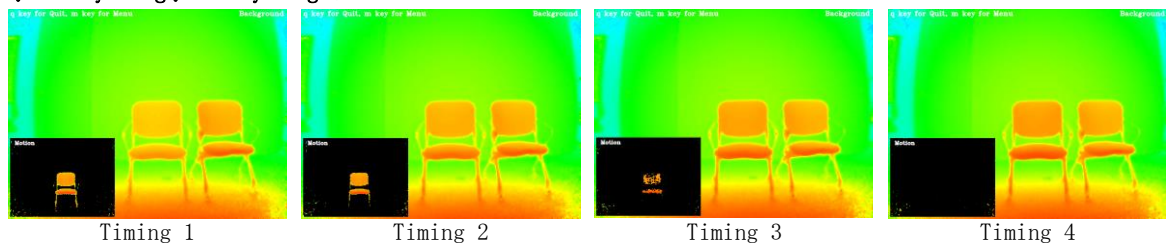
Figure 44 snows changes of motion and background depending on quantity.



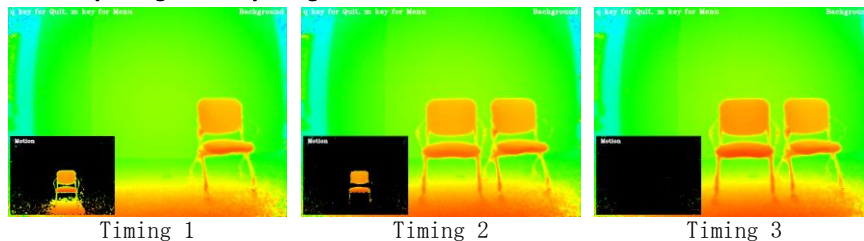
Quantity = BgQuantity::bgLv1



Quantity = BgQuantity::bgLv5



Quantity = BgQuantity::bgLv9



**Figure 39 Sample images of Background Subtraction**

## 5.13.4 Noise Reduction

### 5.13.4.1 Low Signal Cutoff

TOF	TOFv1 or later
SDK	v1.3.0 or later

#### 1) Outline

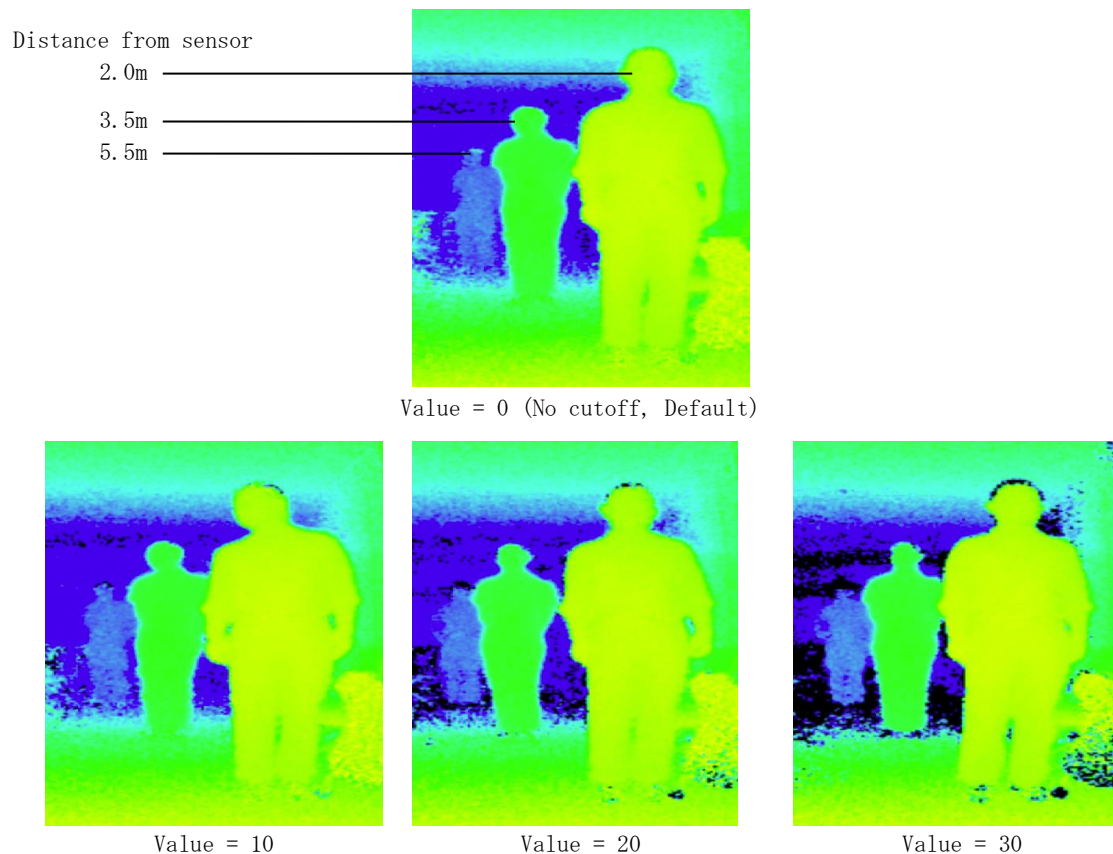
Small signal noise due to multi reflection of emitted laser (Multi-pass) and other noises can be cut off. Pixels which receiving intensity is less than the setting value which is between 0 and 4095, become invalid value (0xFFFF) in order to eliminate abnormal data. Since the larger setting value makes the harder to detect objects in far place and low reflection objects, it is recommended that the setting value is under 30. (Refer to Table 5-13)

**Table 5-13 Low Signal Cutoff**

Operation	Discription	Remark
Tof::SetLowSignalCutoff()	0 : No cutoff(Default) 1 to 4095 : pixels which receiving intensity is less than the setting value is invalid (0xFFFF). 0,10,20,30 are recommended	
Tof::GetLowSignalCutoff()	Get setting value set by SetLowSignalCutoff()	

#### 2) Image Samples

Figure 45 shows sample images of Low Signal Cutoff.



**Figure 40 Sample images of Low Signal Cutoff**

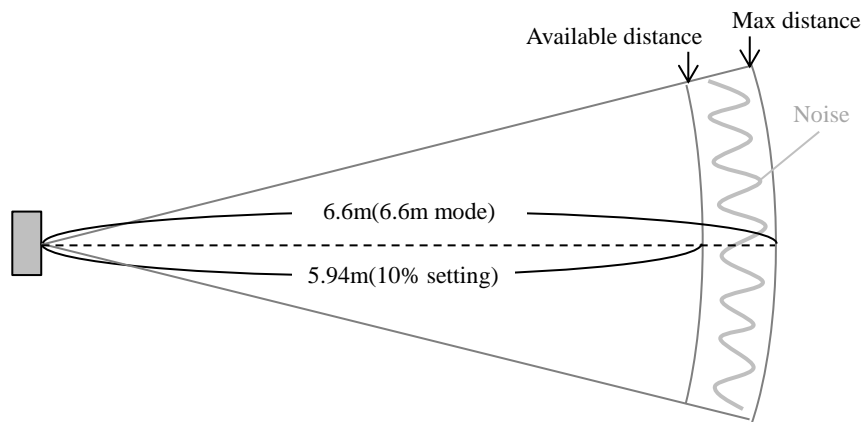


### 5.13.4.2 Far Signal Cutoff

TOF	TOFv1 or later
SDK	v2.1.0 or later

#### 1) Outline

Since it is noisy around far area in a detection range, the far area noise can be removed (disabled) restricting enabled distance if used range is fixed. (Refer to Figure 46)



**Figure 41 Far Signal Cutoff**

Removed range is set as a rate (0.0 to 1.0) against to detection range of the sensor. (Refer to Table 5-14) For instance as Figure 46, 10% of far area is removed by setting value of 0.1, and enabled range is 5.94m that is 90% of 6.6m.

**Table 5-14 Setting of Far Signal Cutoff**

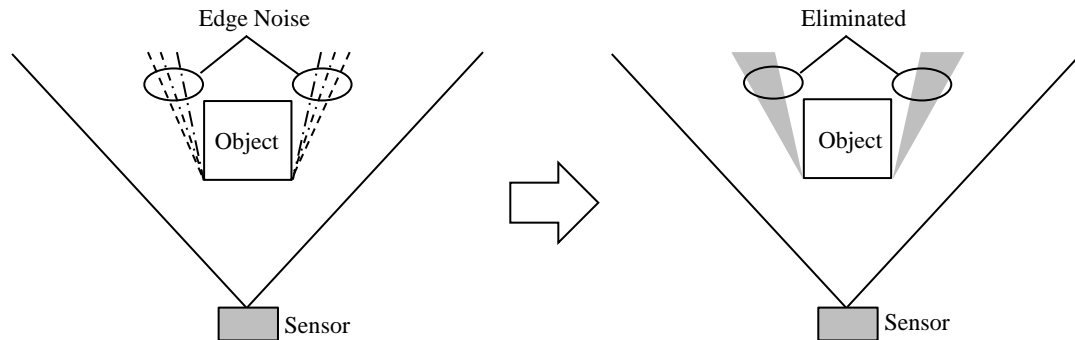
Operation	Description	Remark
Tof::SetFarSignalCutoff()	0.0 to 1.0 : A rate to be cutoff in detection range of the sensor.	
Tof::GetFarSignalCutoff()	Get setting value set by SetFarSignalCutoff()	

### 5.13.4.3 Edge Signal Cutoff

TOF	TOFv1 or later
SDK	v2.1.0 or later

#### 1) Outline

Since laser beam dose not directly return from edges of an object, distance of the edge cannot be measured correctly and noises occur in rear side of the object that is called edge noise. Removing (disabling) outline of an object reduces edge noise. (Refer to Figure 47)



**Figure 42 Edge Signal Cutoff**

This function can be enable and disable by the methods in Table 5-15. The setting can be changed during transferring (Run).

**Table 5-15 Setting of Edge Signal Cutoff**

Operation	Discription	Remark
Tof::SetEdgeSignalCutoff()	EdgeSignalCutoff::Disable : Disabled EdgeSignalCutoff::Enable : Enabled	
Tof::GetEdgeSignalCutoff()	Get setting value set by SetEdgeSignalCutoff()	

## 6 API Reference

### 6.1 TofManager Class

Class to manage all TOF sensors

TofManager

- Create at calling Tof::Open() for Tof instances.
- TofManager instance can be destructed after all Tof instances start to use TOF sensors by Tof::Open().

#### Public Attributes

Name	Type	Description
inifilepath	string	Folder of Initialization File(Default is "", Finish with '/' if it is indicated)
inifilename	string	File name of Initialization File (Default is "tof.ini")
bConsoleLog	bool	Enable/Disable console log by SDK

#### Public Member Functions

Name	Description
Initialize()	Initialize Initialization File (Create as new)
Open()	Start to use TofManager instance (Load Initialization File)
Close()	End to use TofManager instance
GetTofList()	Get TOF information list of all TOF sensors
AddTof()	Add a TOF sensor
DeleteTof()	Delete a TOF sensor

#### Reference

Include : tof.h

### 6.1.1 TofManager::Initialize() Method

Initialize Initialization File (Create as new)

```
Result Initialize(  
                void  
                )
```

- A new Initialization File is created which no TOF sensor is registered.
- If old Initialization File remains, it is overwritten.
- In default, Initialization File is saved to the folder where user application is, and name is "tof.ini".
- infilepath member variable is changed if the save folder is changed. (Finish with file separator '/')
- infilename member variable is changed if the file name of Initialization File is changed.

#### Parameters

None

#### Returns

Result

#### Precondition

- The method shall be called while TofManager instance is in the close state. (Before TofManager::Open() or after TofManager::Close())
- Error is reported if it is called while TofManager instance is in open state.

#### Postcondition

- Initialization File is modified.
- TofManager instance is not yet available with the initialized information even after this method is called until TofManager::Open() method is called.

### 6.1.2 TofManager::Open() Method

Start to use TofManager instance (Load Initialization File)

```
Result Open(  
            void  
            )
```

- Load Initialization File, and change TofManager instance to open state.
- In default, Initialization File is save to the folder where user application is, and name is "tof.ini".
- infilepath member variable is changed if the save folder is changed. (Finish with file separator '/')
- infilename member variable is changed if the file name of Initialization File is changed.

#### Parameters

None

#### Returns

Result

### 6.1.3 TofManager::Close() Method

End to use TofManager instance

```
Result Close(  
            void  
            )
```

- Clear information loaded by TofManager::Open(), change TofManager instance to close state.

#### Parameters

None

#### Returns

Result

### 6.1.4 TofManager::GetTofList() Method

Get TOF information list of all TOF sensors

```
int TofManager::GetTofList(  
                                const TofInfo** toflist  
                                )
```

- Get TOF information list of all TOF sensors which SDK manages.
- Input an element of TOF information list to Tof::Open method of Tof class.

#### Parameters

```
const TofInfo** toflist  
    Pointer to point a Pointer to point TofInfo type array.
```

#### Returns

Number of TOF sensor (-1 is returned if TofManager instance is close state.)

#### Precondition

- TofManager instance shall be valid by TofManager::Open() method in advance.

#### Postcondition

- The output is a pointer to the list created in TofManager instance.  
If TofManager instance is destructed or end to be used by TofManager::Close() instance, contents of the list becomes invalid.

### 6.1.5 TofManager::AddTof() Method

Add a TOF sensor

```
Result TofManager::AddTof(TofMac tofmac,  
                           TofIp tofip,  
                           int rtp_port  
                           )
```

- Add a new TOF sensor information to Initialization File, and SDK will manage it.

#### Parameters

TofMac tofmac  
MAC address of the TOF sensor which is added

TofIp tofip  
IP address of the TOF sensor which is added

int rtp\_port  
RTP port number of the TOF sensor which is added

#### Returns

Result

#### Precondition

- The method shall be called while TofManager instance is in the open state (after TofManager::Open()).
- Error is reported if it is called while TofManager instance is in close state.
- Unique TOF ID is assigned to the newly added TOF sensor.

#### Postcondition

- Initialization File is modified.

### 6.1.6 TofManager::DeleteTof() Method

Delete a TOF sensor

```
Result TofManager::DeleteTof(TofId tofid
                             )
```

- Delete a TOF sensor information in Initialization File, and SDK will not manage it.

#### Parameters

TofId tofid  
TOF ID of the TOF sensor to be deleted

#### Returns

Result

#### Precondition

- Call this method after destruct Tof instance correspond to the TOF sensor.

#### Postcondition

- Initialization File is modified.



## 6.2 Tof Class

Class to control a TOF sensor

Tof

- Set TOF information by Tof::Open() method after creating instance.
- Get TOF information of all TOF sensor managed by SDK through TofManager::GetTofList() method.

### Public Attributes

Name	Type	Description
tofinfo	TofInfo	TOF information of the TOF sensor
capturetime	unsigned int	Max capture duration (seconds). Default is 600 seconds.
bConsoleLog	bool	Enable/Disable console log by SDK

### Public Member Functions

Name	Description
Open ()	Start to use a TOF sensor
Open ()	Start to use an Emulated TOF sensor
Close ()	End to use the TOF sensor
Run ()	Start transferring from the TOF sensor
Run ()	Start transferring from the TOF sensor (for Human Detect)
Stop ()	Stop transferring from the TOF sensor
Restart ()	Restart the TOF sensor
SetStandbyMode ()	Set Standby Mode for the TOF sensor
GetVersion ()	Get versions of each module in the TOF sensor
GetStatus ()	Get status of the TOF sensor
GetFrameStatus ()	Get status of the latest received Frame
ReadFrame ()	Read the latest received Depth Frame data
ReadFrame ()	Read two latest received Depth Frame data simultaneously
ReadFrame ()	Read the latest received Depth and IR data Frames simultaneously
ReadFrame ()	Read the latest received IR data Frame
ReadFrame ()	Read the latest received Depth Frame data and convert to 3D
ReadFrame ()	Result of human detection in the latest frame
SetCameraMode ()	Set Camera Mode (Depth/IR)
GetCameraMode ()	Get the current setting of Camera Mode (Depth/IR)
SetCameraPixel ()	Set Pixel size of an image
GetCameraPixel ()	Get the current setting of Pixel size of an image
SetIrGain ()	Set Exposure Sensitivity for IR Image
GetIrGain ()	Get the current setting of Exposure Sensitivity for IR Image
SetDistanceMode ()	Set Distance Mode of the TOF sensor
GetDistanceMode ()	Get the current setting of Distance Mode of the TOF sensor
SetFrameRate ()	Set Frame Rate of the TOF sensor
GetFrameRate ()	Get the current setting of Frame Rate of the TOF sensor
SetLowSignalCutoff ()	Set low signal cutoff
GetLowSignalCutoff ()	Get low signal cutoff
SetFarSignalCutoff ()	Set far signal cutoff
GetFarSignalCutoff ()	Get far signal cutoff
SetEdgeSignalCutoff ()	Set edge signal cutoff
GetEdgeSignalCutoff ()	Get edge signal cutoff

ResetBackground()	Initialize background(Reset)
SetBackgroundInterval()	Set interval to update background
GetBackgroundInterval()	Get the current setting of interval to update background
SetBackgroundQuantity()	Set quantity to update background
GetBackgroundQuantity()	Get the current setting of quantity to update background
CreateCaptureFile()	Create capture file
Capture()	Start or stop capturing
GetCaptureStatus()	Get status of capturing
PauseEmulationTof()	Pause/Resume of play
SetAttribute()	Set physical installation position of TOF sensor
GetAttribute()	Get physical installation position of TOF sensor

## Reference

Include : tof.h

### 6.2.1 Tof::Open() Method

Start to use a TOF sensor

```
Result Tof::Open(TofInfo tofinfo
                )
```

- Invalid to access to TOF sensor until this method is called even after Tof instance is created.
- Input TOF information of a TOF sensor gotten through TofManager::GetTofList() as an argument.
- If TOF sensor is in Standby mode, Standby mode is canceled. (from v1.1.0)

#### Parameters

TofInfo tofinfo  
TOF information of a TOF to be started

#### Returns

Result

### 6.2.2 Tof::Open() Method

Start to use an Emulated TOF sensor

```
Result Tof::Open(CaptureInfo info
                )
```

- Start Tof instance in emulation mode to replay a capture file.
- Possible to start without connecting real TOF sensor.
- Settings for TOF sensor is according to the capture file.
- Functions other than replaying capture file are basically invalid.

#### Parameters

CaptureInfo info  
Full path of capture file + file name (Folder separator is /)

#### Returns

Result

#### Note

- Available from v1.3.0.

### 6.2.3 Tof::Close() Method

End to use the TOF sensor

```
Result Tof::Close(void  
)
```

- Disconnected with a TOF sensor in this method.

#### Parameters

None

#### Returns

Result

### 6.2.4 Tof::Run() Method

Start transferring from the TOF sensor

```
Result Tof::Run(void  
)
```

- RTP transferring from TOF sensor and start to receive Frame data.

#### Parameters

None

#### Returns

Result

### 6.2.5 Tof::Run() Method

Start transferring from the TOF sensor (for Human Detect)

```
Result Tof::Run(RunMode mode  
                )
```

- In case of RunMode::Normal, Start RTP transferring from TOF sensor and start to receive Frame data.
- In case of RunMode::HumanDetect, Start RTP transferring from TOF sensor and human detection is started in background.
- In case of RunMode::HumanDetect, installed position of TOF sensor must be set by SetAttribute() method in advance.

#### Parameters

RunMode mode  
Operation with receiving data.

#### Returns

Result

#### Note

- Available from v1.3.0.

### 6.2.6 Tof::Stop() Method

Stop transferring from the TOF sensor

```
Result Tof::Stop(void  
                  )
```

- Stop RTP transferring from TOF sensor and stop to receive Frame data.

#### Parameters

None

#### Returns

Result

### 6.2.7 Tof::Restart() Method

Restart the TOF sensor

```
Result Tof::Restart(void  
    )
```

- Restart and initialize the TOF sensor.
- Execute this method to restart the TOF sensor when TOF sensor does not reply properly or there is no response from TOF sensor. As in case of network trouble, could not restart TOF sensor by this method, check network status or restart TOF sensor by disconnecting and connecting the cable.
- It takes 30 seconds to several minutes for Restart the TOF sensor.

#### Parameters

None

#### Returns

Result

### 6.2.8 Tof::SetStandbyMode() Method

Set Standby Mode for the TOF sensor

```
Result Tof::SetStandbyMode(bool enabled  
    )
```

- If the TOF sensor will not be used for long time, set Standby mode in order to reduce power consumption.
- Since it is automatically set in Standby mode during stop status, this method is not necessary to be used.

#### Parameters

bool enabled  
Mode (true: Standby Mode false: Cancel)

#### Returns

Result

#### Precondition

- It shall be set in stopping status before running by Tof::Run() or after stopping by Tof::Stop().
- Error is returned if it is set during transferring.

### 6.2.9 Tof::GetVersion() Method

Get versions of each module in the TOF sensor

```
Result Tof::GetVersion(string* fpga,  
                      string* osvers,  
                      string* root  
                      )
```

- Get versions of each module in the TOF sensor.

#### Parameters

string\* fpga  
    Pointer to output FPGA version

string\* osvers  
    Pointer to output LinuxOS version

string\* root  
    Pointer to output TOF sensor firmware version

#### Returns

Result

### 6.2.10 Tof::GetStatus() Method

Get status of the TOF sensor

```
Result Tof::GetStatus(unsigned char* status  
                      )
```

- Get the latest status of the TOF sensor.

#### Parameters

unsigned char\* status  
    Pointer to output status

#### Returns

Result

### 6.2.11 Tof::GetFrameStatus() Method

Get status of the latest received Frame

```
Result Tof::GetFrameStatus(long* frameno,  
                           TimeStamp* timestamp  
                           )
```

- Get the latest Frame information which SDK received.
- If Frame number or Timestamp is different from previous recieved data, it means a new Frame was received.
- In case of emulation mode, frameno is reported as -2 at the end of capture file.

#### Parameters

long\* frameno

Frame Number (0 to 0x7fffffff(Round every about 820 days)) of the latest Frame, -1 is set if not received yet)

TimeStamp\* timestamp

Timestamp(UTC)

#### Returns

Result

#### Postcondition

- The latest Frame can be read calling ReadFrame() after Frame number or Timestamp is changed.
- The newer Frame than this method reported may be read by ReadFrame() method depending on timing.



### 6.2.12 Tof::ReadFrame() Method

Read the latest received Depth Frame data

```
Result Tof::ReadFrame(FrameDepth* frame
                      )
```

- Read the latest Depth Frame data into indicated FrameDepth instance.

#### Parameters

FrameDepth\* frame  
FrameDepth instance to be read into

#### Returns

Result

#### Note

- This method waits until the first Frame is received.
- If a newer Frame than the Frame in inputted FrameDepth instance is not received yet, wait until a new Frame arrival.

#### Precondition

- Create FrameDepth instance in user application.
- Set Camera Mode in Depth Mode in advance.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

### 6.2.13 Tof::ReadFrame() Method

Read two latest received Depth Frame data simultaneously

```
Result Tof::ReadFrame(FrameDepth* frame1,  
                      FrameDepth* frame2  
                      )
```

- Read two latest Depth Frame data simultaneously into indicated FrameDepth instance. Refer to FrameData(5.8) about two depth frame data.

#### Parameters

FrameDepth\* frame1  
FrameDepth instance1 to be read into

FrameDepth\* frame2  
FrameDepth instance2 to be read into

#### Returns

Result

#### Note

- This method waits until the first Frame is received.
- If a newer Frame than the Frame in inputted FrameDepth instance is not received yet, wait until a new Frame arrival.
- Available from v2.0.0 or later.

#### Precondition

- Create FrameDepth instance in user application.
- Set Camera Mode in Depth Mode/Motion/Background dual output in advance.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

## 6.2.14 Tof::ReadFrame() Method

Read the latest received Depth and IR data Frames simultaneously

```
Result Tof::ReadFrame(FrameDepth* frame1,  
                      FrameIr* frame2  
                      )
```

- Read the latest Depth Frame data into indicated FrameDepth instance.
- Read the latest IR Frame data into indicated FrameIr instance.

### Parameters

FrameDepth\* frame1  
FrameDepth instance1 to be read into

FrameIr \* frame2  
FrameIr instance to be read into

### Returns

Result

### Note

- This method waits until the first Frame is received.
- If a newer Frame than the Frame in inputted FrameDepth instance is not received yet, wait until a new Frame arrival.
- Available from v2.0.0 or later.

### Precondition

- Create FrameDepth instance in user application.
- Set Camera Mode in Depth Mode/Motion/Background and IR dual output in advance.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

### 6.2.15 Tof::ReadFrame() Method

Read the latest received IR data Frame

```
Result Tof::ReadFrame(FrameIr* frame
                      )
```

- Read the latest IR Frame data into indicated FrameIr instance.

#### Parameters

FrameIr\* frame  
FrameIr instance to be read into

#### Returns

Result

#### Precondition

- Create FrameIr instance in user application.
- Set Camera Mode in IR Mode in advance.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

## 6.2.16 Tof::ReadFrame() Method

Read the latest received Depth Frame data and convert to 3D

```
Result Tof::ReadFrame(Frame3d* frame
                      )
```

- Read the latest 3D Frame data into indicated Frame3d instance after lens correction and conversion from depth data.

### Parameters

Frame3d\* frame  
Frame3d instance to be read into

### Returns

Result

### Precondition

- Create Frame3d instance in user application.
- Set Camera Mode in Depth Mode in advance.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

### Note

- This method waits until the first Frame is received.
- If a newer Frame than the Frame in inputted FrameDepth instance is not received yet, wait until a new Frame arrival.
- Available from v1.1.0.

### 6.2.17 Tof::ReadFrame() Method

Result of human detection in the latest frame

```
Result Tof::ReadFrame(FrameHumans* frame
                      )
```

- Read the latest result of human detection in the latest frame to FrameHumans instance.

#### Parameters

FrameHumans\* frame  
FrameHumans instance to be read into

#### Returns

Result

#### Precondition

- Set RunMode::HumanDetect mode in Run method.
- Create FrameHumans instance in user application.
- It is recommended to call this method after confirming received Frame is changed by GetFrameStatus().

#### Note

- This method waits until the first Frame is received.
- If a newer Frame than the Frame in inputted FrameDepth instance is not received yet, wait until a new Frame arrival.
- Available from v1.3.0.

## 6.2.18 Tof::SetCameraMode() Method

Set Camera Mode

```
Result Tof::SetCameraMode(CameraMode mode
                           )
```

- Set Camera Mode to the TOF sensor.
- Refer to CameraMode (6.10.1) for setting values.

### Parameters

CameraMode mode  
Camera Mode

### Returns

Result

### Precondition

- It shall be set in stopping status before running by Tof::Run() or after stopping by Tof::Stop().
- Error is returned if it is set during transferring.

## 6.2.19 Tof::GetCameraMode() Method

Get the current setting of Camera Mode

```
Result Tof::GetCameraMode(CameraMode* mode
                           )
```

- Get the current setting of Camera Mode of the TOF sensor.
- Refer to CameraMode (6.10.1) for setting values.

### Parameters

CameraMode\* mode  
Pointer to output Camera Mode setting

### Returns

Result

### 6.2.20 Tof::SetCameraPixel() Method

Set Pixel size of an image

```
Result Tof::SetCameraPixel(CameraPixel pixel
                           )
```

- Change Pixel size of an image of the TOF sensor.
- Refer to CameraPixel (6.10.3) for setting values.

#### Parameters

CameraPixel pixel  
Pixel size mode

#### Returns

Result

#### Precondition

- It shall be set in stopping status before running by Tof::Run() or after stopping by Tof::Stop().
- Error is returned if it is set during transferring.

### 6.2.21 Tof::GetCameraPixel() Method

Get the current setting of Pixel size of an image

```
Result Tof::GetCameraPixel(CameraPixel* pixel
                           )
```

- Get the current setting of Pixel size of the TOF sensor.
- Refer to CameraPixel (6.10.3) for setting values.

#### Parameters

CameraPixel\* pixel  
Pointer to output Pixel size setting

#### Returns

Result



### 6.2.22 Tof::SetIrGain() Method

Set Exposure Sensitivity for IR Image

```
Result Tof::SetIrGain(int irgain  
                      )
```

- Exposure Sensitivity for IR can be adjusted if IR image is too dark or bright due to surrounding light.

#### Parameters

int irgain  
Gain setting value for IR image (set from 1 to 15. Default is 8.)

#### Returns

Result

#### Precondition

- This setting can be changed during transferring.

### 6.2.23 Tof::GetIrGain() Method

Get the current setting of Exposure Sensitivity for IR Image

```
Result Tof::GetIrGain(int* irgain  
                      )
```

- Get the current setting of Exposure Sensitivity for IR Image.

#### Parameters

int\* irgain  
Pointer to output Gain setting (1 to 15) for IR image

#### Returns

Result

### 6.2.24 Tof::SetDistanceMode() Method

Set Distance Mode of the TOF sensor

```
Result Tof::SetDistanceMode(DistanceMode mode  
                             )
```

- Set Distance Mode of the TOF sensor.
- Refer to DistanceMode (6.10.4) for setting values.

#### Parameters

DistanceMode mode  
Distance mode

#### Returns

Result

### 6.2.25 Tof::GetDistanceMode() Method

Get the current setting of Distance Mode of the TOF sensor

```
Result Tof::GetDistanceMode(DistanceMode* mode  
                             )
```

- Get the current setting of Distance Mode of the TOF sensor.
- Refer to DistanceMode (6.10.4) for setting values.

#### Parameters

DistanceMode\* mode  
Distance mode

#### Returns

Result

### 6.2.26 Tof::SetFrameRate() Method

Set Frame Rate of the TOF sensor

```
Result Tof::SetFrameRate(FrameRate fps
                          )
```

- Set Frame Rate of the TOF sensor.
- Refer to FrameRate (6.10.5) for setting values.

#### Parameters

FrameRate fps  
Frame rate

#### Returns

Result

#### Note

- Not available yet in v1.2.0.
- TOF sensor which supports the function is required.

### 6.2.27 Tof::GetFrameRate() Method

Get the current setting of Frame Rate of the TOF sensor

```
Result Tof::GetFrameRate(FrameRate* fps
                          )
```

- Get the current setting of Frame Rate of the TOF sensor.
- Refer to FrameRate (6.10.5) for setting values.

#### Parameters

FrameRate\* fps  
Frame rate

#### Returns

Result

#### Note

- Not available yet in v1.2.0.
- TOF sensor which supports the function is required.

## 6.2.28 Tof::SetLowSignalCutoff() Method

Set low signal cutoff

```
Result Tof::SetLowSignalCutoff(unsigned int value
                                )
```

- Set signal level to be cutoff.
- Noise near by or far from sensor can be cutoff.
- If cutoff level is too strong, normal signal may be also cutoff.
- 0(No cutoff) to 30 is recommended value.

### Parameters

unsigned int value

Signal level to cutoff (0 to 4095). Larger value is larger cutoff level.

### Returns

Result

### Note

- It is possible to change during Run() of TOF.
- Available from v1.3.0.

## 6.2.29 Tof::GetLowSignalCutoff() Method

Get low signal cutoff

```
Result Tof::GetLowSignalCutoff(unsigned int* value
                                )
```

- Get signal level to be cutoff.

### Parameters

unsigned int\* value

Signal level to cutoff (0 to 4095). Larger value is larger cutoff level.

### Returns

Result

### Note

- It is possible to get during Run() of TOF.
- Available from v1.3.0.

### 6.2.30 Tof::SetFarSignalCutoff() Method

Set far signal cutoff

```
Result Tof::SetFarSignalCutoff(float rate  
                                )
```

- Set distance rate far from TOF.

#### Parameters

unsigned int value  
Distance rate far from TOF (rate: 0.0 ~ 1.0)

#### Returns

Result

#### Note

- rate = 0 is no cutoff.
- It is possible to change during Run() of TOF.
- Available from v2.1.0.

### 6.2.31 Tof::GetFarSignalCutoff() Method

Get far signal cutoff

```
Result Tof::GetFarSignalCutoff(float* rate  
                                )
```

- Get distance rate far from TOF.

#### Parameters

unsigned int\* value  
Distance rate far from TOF (rate: 0.0 ~ 1.0)

#### Returns

Result

#### Note

- rate = 0 is no cutoff.
- It is possible to get during Run() of TOF.
- Available from v2.1.0.

### 6.2.32 Tof::SetEdgeSignalCutoff() Method

Set edge noise reduction mode

```
Result Tof::SetEdgeSignalCutoff(EdgeSignalCutoff mode
                                )
```

- Reduce edge noise.

#### Parameters

EdgeSignalCutoff mode  
edge noise reduction mode

#### Returns

Result

#### Note

- This setting can be changed during transferring.
- Available from v2.1.0.

### 6.2.33 Tof::GetEdgeSignalCutoff() Method

Get far signal cutoff

```
Result Tof::GetEdgeSignalCutoff(EdgeSignalCutoff* mode
                                )
```

- Get edge noise reduction mode

#### Parameters

EdgeSignalCutoff\* mode  
edge noise reduction mode

#### Returns

Result

#### Note

- This setting can be changed during transferring.
- Available from v2.1.0.

## 6.2.34 Tof::ResetBackground() Method

Initialize background(Reset)

```
Result Tof::ResetBackground(void  
    )
```

- Initialize background(Reset).

### Parameters

None

### Returns

Result

### Note

- Available from v2.0.0.

### 6.2.35 Tof::SetBackgroundInterval() Method

Set interval to update background

```
Result Tof::SetBackgroundInterval(BgInterval interval
)
```

- Set interval to update background.

#### Parameters

BgInterval interval

Refer to BgInterval (#6.10.7) for setting values.

#### Returns

Result

#### Note

- It is possible to set during Run() of TOF
- Available from v2.0.0.

### 6.2.36 Tof::GetBackgroundInterval() Method

Get the current setting of interval to update background.

```
Result Tof::GetBackgroundInterval(BgInterval* interval
)
```

- Get the current setting of interval to update background.

#### Parameters

BgInterval\* interval

Refer to BgInterval (#6.10.7) for setting values.

#### Returns

Result

#### Note

- Available from v2.0.0.



### 6.2.37 Tof::SetBackgroundQuantity() Method

Set quantity to update background

```
Result Tof::SetBackgroundQuantity(BgQuantity quantity
                                   )
```

- Set quantity to update background.

#### Parameters

BgQuantity quantity

Refer to BgQuantity (#6.10.8) for setting values.

#### Returns

Result

#### Note

- It is possible to set during Run() of TOF
- Available from v2.0.0.

### 6.2.38 Tof::GetBackgroundQuantity() Method

Get the current setting of quantity to update background

```
Result Tof::GetBackgroundQuantity(BgQuantity* quantity
                                   )
```

- Get the current setting of quantity to update background.

#### Parameters

BgQuantity\* quantity

Refer to BgQuantity (#6.10.8) for setting values.

#### Returns

Result

#### Note

- Available from v2.0.0.

### 6.2.39 Tof::CreateCaptureFile() Method

Create capture file

```
Result Tof::CreateCaptureFile(CaptureInfo info  
                               )
```

- Create capture file.

#### Parameters

CaptureInfo info

Path to save. (Should use half-width character and finish with folder separator (/))

If it is blank, same path with .exe is set.

#### Returns

Result

#### Note

- If already existing file is inputted, it will be overwritten.
- Available from v1.3.0.

## 6.2.40 Tof::Capture() Method

Start or stop capturing

```
Result Tof::Capture(bool bEnable  
                    )
```

- Start or stop capturing.
- Capturing only if capture is started and it is in Run status.

### Parameters

bool bEnable  
true: Start capturing.  
false: Stop capturing.

### Returns

Result

### Note

- Don't change settings for TOF sensor during capture is stopped.
- Start again from CreateCaptureFile() if any setting for TOF sensor is changed.
- Available from v1.3.0.

## 6.2.41 Tof::GetCaptureStatus() Method

Get status of capturing

```
Result Tof::GetCaptureStatus(CaptureStatus* status  
                             )
```

- Get status of capturing.

### Parameters

CaptureStatus\* status  
Refer to CaptureStatus for capture status.

### Returns

Result

### Note

- Available from v1.3.0.

## 6.2.42 Tof::PauseEmulationTof() Method

Pause/Resume of play

```
Result Tof::PauseEmulationTof(bool bPause
                               )
```

- Pause/Resume of play.

### Parameters

bool bPause  
true: Pause  
false: Resume

### Returns

Result

### Note

- Available from v2.0.0.

### 6.2.43 Tof::SetAttribute() Method

Set physical installation position of TOF sensor for human detection.

```
Result Tof::SetAttribute(float x,  
                        float y,  
                        float z,  
                        float rx,  
                        float ry,  
                        float rz,  
                        )
```

- Set physical installation position of TOF sensor.
- 3D rotation order is, Z-axis, Y-axis, and X-axis.

#### Parameters

float x

x-coordinate of TOF sensor [mm]

float y

y-coordinate of TOF sensor [mm]

float z

z-coordinate of TOF sensor(-1 \* height from floor) [mm]

float rx

X-axis rotation angle(0 to 359 degree)

float ry

Y-axis rotation angle(0 to 359 degree)

float rz

Z-axis rotation angle(0 to 359 degree)

#### Returns

Result

#### Note

- Available from v1.3.0.

### 6.2.44 Tof::GetAttribute() Method

Get physical installation position of TOF sensor for human detection.

```
Result Tof::GetAttribute(float* x,  
                        float* y,  
                        float* z,  
                        float* rx,  
                        float* ry,  
                        float* rz,  
                        )
```

- Get physical installation position of TOF sensor.

#### Parameters

float\* x  
x-coordinate of TOF sensor [mm]

float\* y  
y-coordinate of TOF sensor [mm]

float\* z  
z-coordinate of TOF sensor(-1 \* height from floor) [mm]

float\* rx  
X-axis rotation angle(0 to 359 degree)

float\* ry  
Y-axis rotation angle(0 to 359 degree)

float\* rz  
Z-axis rotation angle(0 to 359 degree)

#### Returns

Result

#### Note

- Available from v1.3.0.

## 6.3 FrameData Class

Abstract class to store a Frame Data

Cannot be used due to abstract class

### Public Attributes

Name	Type	Description
tofinfo	TofInfo	TOF sensor information which the data was gotten
timestamp	TimeStamp	Timestamp(UTC)
framenumbers	long	Frame Number (0 to 0x7ffffff(Round every about 820 days))
modelname	string	Model name of TOF sensor which the data was gotten
distance_min	float	Minimum measurable distance [mm] of the data (Actual distance for 0x0000 of data)
distance_max	float	Maximum measurable distance [mm] of the data (Actual distance for 0x0000 of data)
lens	LensParam	Lens correction information

### Public Member Functions

Name	Description

### Reference

Include : tof.h

## 6.4 FrameMatrix Class

Abstract class to store a Frame of Depth or IR Matrix data

### Base Class

FrameData

Cannot be used due to abstract class

### Public Attributes

Name	Type	Description
width	int	Width of Matrix
height	int	Height of Matrix
picbyte	int	Byte size of a pixel
pixel	int	Total pixels
databuf	std::vector< WORD >	Matrix data (16 bit (little endian) x pixels)

### Public Member Functions

Name	Description

### Reference

Include : tof.h



## 6.5 FrameDepth Class

Class to store a Frame of Depth data

### Base Class

FrameData, FrameMatrix

FrameDepth

- Use instance of this class to read Frame data in Depth mode.
- Data (16 bit) of each pixel is 0x0000 to 0xfffe. (0xffff is invalid data)
- To convert the data to actual distance between the object, it should be calculated as 0x0000 = distance\_min member, 0xfffe = distance\_max member. (both members are defined in FrameData class.) The unit is [mm].  
(from v1.1.0, it is possible by CalculateLength() method also.)
- A pixel data (16bit) is stored as little endian from v1.1.0.

### Public Attributes

Name	Type	Description
ColorTable[]	unsigned char	Color table

### Public Member Functions

Name	Description
CreateColorTable()	Create color table to display Dept data as a color image
Save()	Save to file
CalculateLength()	Calculate actual length

### Reference

Include : tof.h

### 6.5.1 FrameDepth::CreateColorTable() Method

Create color table to display Dept data as a color image

```
void FrameDepth::CreateColorTable(unsigned short scaleMin,  
                                  unsigned short scaleRange  
                                  )
```

- Set color table to output to JPEG/PNG file.

#### Parameters

unsigned short scaleMin  
Min value of Scale

unsigned short scaleRange  
Range of Scale

#### Returns

None

### 6.5.2 FrameDepth::Save() Method

Save to file

```
Result FrameDepth::Save(char* filename,  
                        OutputDataType type  
                        )
```

- Convert Matrix data to selected data type and save to file.

#### Parameters

char\* filename  
Output file name

OutputDataType type  
Output data type

#### Returns

Result

#### Precondition

- If save as JPEG/PNG, set color table by FrameDepth::CreateColorTable() method in advance.

#### Note

- Save as Binary is available from v1.1.0. Save as JPEG/PNG is available from v1.2.0.
- If filename is empty, file name is automatically allocated based on Timestamp.

### 6.5.3 FrameDepth::CalculateLength() Method

Calculate actual length

```
float FrameDepth::CalculateLength(unsigned short depth
                                )
```

- Convert depth data to actual length (mm).

#### Parameters

unsigned short depth  
Depth data read by Tof::ReadFrame() method

#### Returns

float  
Length (mm)  
-1 is returned for invalid data (0xFFFF)

#### Note

- Available from v1.1.0.

## 6.6 FrameIr Class

Class to store a Frame of IR data

### Base Class

FrameData, FrameMatrix

```
FrameIr
```

- Use instance of this class to read Frame data in IR mode.

### Public Attributes

Name	Type	Description

### Public Member Functions

Name	Description
Save ()	Save to file

### Note

- At v1.1.0, IR data is set as 12bit data (0 to 4095). From v1.2.0, IR data is set as 16bit data (0 to 65535).

### Reference

Include : tof.h

### 6.6.1 FrameIr::Save() Method

Save to file

```
Result FrameIr::Save(char* filename,  
                    OutputDataType type  
                    )
```

- Convert Matrix data to selected data type and save to file.

#### Parameters

char\* filename  
Output file name

OutputDataType type  
Output data type

#### Returns

Result

#### Precondition

- No need to set color table because IR image is black & white.

#### Note

- Save as Binary is available from v1.1.0. Save as JPEG/PNG is available from v1.2.0.
- If filename is empty, file name is automatically allocated based on Timestamp.

## 6.7 Frame3d Class

Class to store a Frame of 3D data

### Base Class

FrameData

```
Frame3d
```

- Use instance of this class to read Frame data in 3D mode.
- 3D coordinate of each point corresponding to pixel is represented as x,y,z- coordinate.
- 3D frame data can be directly read by `Tof::ReadFrame(Frame3d*)`.
- Depth frame data can be converted to 3D by `Frame3d::Convert(FrameDepth*)` method.

### Note

- Available from v1.1.0.

### Public Attributes

Name	Type	Description
width	int	Width of Matrix
height	int	Height of Matrix
pixel	int	Total pixels
frame3d	std::vector<TofPoint>	Coordinate data

### Public Member Functions

Name	Description
Convert()	Convert depth data to 3D coordinate
Rotate()	Rotate 3D point data (X,Y,Z order)
RotateZYX()	Rotate 3D point data (Z,Y,X order)

### Reference

Include : `tof.h`

### 6.7.1 Frame3d::Convert() Method

Convert depth data to 3D coordinate

```
Result Frame3d::Convert(FrameDepth* frame
                        )
```

- Convert depth data to 3D point data after lens correction based on lens correction information.

#### Parameters

FrameDepth\* frame

FrameDepth instance which has frame data read by `Tof::ReadFrame(FrameDepth*)`.

#### Returns

Result

#### Note

- Available from v1.1.0.

### 6.7.2 Frame3d::Rotate() Method

Rotate 3D point data (X,Y,Z order)

```
Result Rotate(float rx,
              float ry,
              float rz,
              )
```

- Rotate all 3D point data by specified angle.
- Rotate direction is clockwise toward the positive direction of each axis.
- Rotation order is, X-axis rotation, Y-axis rotation, and Z-axis rotation

#### Parameters

float rx

X-axis rotate angle (0 to 360 degree)

float ry

Y-axis rotate angle (0 to 360 degree)

float rz

Z-axis rotate angle (0 to 360 degree)

#### Returns

Result

#### Note

- Available from v1.3.0.

### 6.7.3 Frame3d::RotateZYX() Method

Rotate 3D point data (Z,Y,X order)

```
Result Rotate(float rx,  
              float ry,  
              float rz,  
              )
```

- Rotate all 3D point data by specified angle.
- Rotate direction is clockwise toward the positive direction of each axis.
- Rotation order is, Z-axis rotation, Y-axis rotation, and X-axis rotation

#### Parameters

float rx  
X-axis rotate angle (0 to 360 degree)

float ry  
Y-axis rotate angle (0 to 360 degree)

float rz  
Z-axis rotate angle (0 to 360 degree)

#### Returns

Result

#### Note

- Available from v2.1.0.



## 6.8 FrameHumans Class

Class to store a Frame of human detection data

### Base Class

FrameData

```
FrameHumans
```

- Use instance of this class to read Frame data of human detection.

### Note

- Available from v1.3.0.

### Public Attributes

Name	Type	Description
humans	<code>std::vector&lt;Human&gt;</code>	Array of human information for detected humans.
numofhuman	<code>int</code>	Number of detected humans.
z_max	<code>float</code>	Z-coordinate of upper limit of detection range[mm] (Negative value as floor level is 0mm)
z_min	<code>float</code>	Z-coordinate of lower limit of detection range[mm] (Negative value as floor level is 0mm)

### Public Member Functions

Name	Description

### Reference

Include : `tof.h`

## 6.9 Structure

### 6.9.1 TofInfo Structure

Structure of TOF Information of a TOF sensor

Member	Type	Description
tofver	unsigned int	TOF version (TOFv1 is 1, TOFv2 is 2)
tofid	TofId (string)	TOF ID (It is allocated at production of TOF sensor and never be changed)
tofmac	TofMac (string)	MAC address
tofigp	TofIp (string)	IP address
rtp_port	int	RTP port number
distance_min	float	Minimum measurable distance [mm] of the data (Actual distance for 0x0000 of data)
distance_max	float	Maximum measurable distance [mm] of the data (Actual distance for 0xfffe of data)

### 6.9.2 TimeStamp Structure

Structure of timestamp

- Timestamp (UTC)

Member	Type	Description
year	unsigned short	Year
month	unsigned short	Month
dayofweek	unsigned short	Weekday(Sun=0, Mon=1, Tue=2, Wed=3, Thu=4, Fri=5, Sat=6)
day	unsigned short	Day
hour	unsigned short	Hour
minute	unsigned short	Minute
second	unsigned short	Second
msecond	unsigned short	Millisecond

### 6.9.3 LensParam Structure

Lens information of the TOF sensor which got the frame data

Member	Type	Description
focallength	float	Focal length of TOF sensor which the data was gotten
fov_x	float	Horizontal FOV (Degree) of TOF sensor which the data was gotten
fov_y	float	Vertical FOV (Degree) of TOF sensor which the data was gotten
ellipticity	float	Ellipticity of lens of TOF sensor which the data was gotten
distortion [NUM_OF_DISTORTION]	double	Distortion of lens of TOF sensor which the data was gotten
shading[NUM_OF_SHADING]	double	Shading correction data

### 6.9.4 CaptureInfo Structure

Structure to set capture file

Member	Type	Description
path	string	Path of capture file (Finish with file separator '/')
filename	string	Name of capture file

### 6.9.5 TofPoint Structure

3D point coordinate

Member	Type	Description
x	float	x-coordinate
y	float	y-coordinate
z	float	z-coordinate

### 6.9.6 Human Structure

Structure of human detect information

Member	Type	Description
id	long	Human ID
x	float	X-coordinate of human [mm]
y	float	Y-coordinate of human [mm]
direction	float	Direction of movement, or direction hand if hand is detected. [0 to 360 degree]
headheight	float	Height of human head [mm] (0 means invalid)
handheight	float	Height of hand if hand is detected [mm] (0 means invalid)
status	HumanStatus	Status of human

## 6.10 Enumerated Type

### 6.10.1 RunMode Type

Run Mode

- Set for Run() method of Tof class.

Name	Value	Description
Normal	0	Normal Mode
HumanDetect	1	Human Detect Mode
FrameEmulation	2	Frame by frame play in emulation mode
Unknown	-1	Not set or Unknown Mode

### 6.10.2 CameraMode Type

Camera Mode

- Set through SetCamera() method of Tof class.
- Motion, Background and 2 frames simultaneous output is available from v2.0.0.

Name	Value	Description
CameraModeDepth	0	Depth Mode
CameraModeIr	1	IR Mode
CameraModeMotion	2	Motion
CameraModeBackground	3	Background
Depth_Motion	4	Depth and Motion
Depth_Background	5	Depth and Background
Depth_Ir	6	Depth and IR
Motion_Background	7	Motion and Background
Motion_Ir	8	Motion and IR
Background_Ir	9	Background and IR
CameraModeUnknown	-1	Not set or Unknown Mode

### 6.10.3 CameraPixel Type

To set pixels of a Camera Image

- Set through SetCameraPixel() method of Tof class.

Name	Value	Description
w640h480	0	640 x 480(VGA)
w320h240	1	320 x 240(QVGA)
w160h120	2	160 x 120(QQVGA)
w80h60	3	80 x 60
w64h48	4	64 x 48
w40h30	5	40 x 30
w32h24	6	32 x 24

### 6.10.4 DistanceMode Type

To set Distance Mode

- Set through SetDistanceMode() method of Tof class.

Name	Value	Description
dm 0 5x	0	0.5x: Half Distance of Default Distance Mode
dm 1 0x	1	1.0x: Default Distance Mode
dm 1 5x	2	1.5x: One and a half Distance of Default Distance Mode
dm 2 0x	3	2.0x: Twice Distance of Default Distance Mode
Unknown	-1	Not set or Unknown Mode

### 6.10.5 FrameRate Type

To set frame rate

- Set through SetFrameRate(), GetFrameRate() method of Tof class.

Name	Value	Description
fr30fps	0	30 fps
fr16fps	1	16 fps
fr10fps	2	10 fps
fr8fps	3	8 fps
fr4fps	4	4 fps
fr2fps	5	2 fps
fr1fps	6	1 fps

#### Note

- TOF sensor which supports the function is required.

### 6.10.6 OutputData Type

To set save file type

- Set through Save() method of FrameData class and FrameIr class.

Name	Value	Description
BinFile	0	Binary File
PngFile	1	PNG File
JpgFile	2	JPG File

### 6.10.7 BgInterval Type

To set interval to update background

- Set through SetBackgroundInterval(), GetBackgroundInterval() method of Tof class.

Name	Value	Description
bg1min	1	1 minute
bg3min	2	3 minutes
bg5min	3	5 minutes
bg10min	4	10 minutes
bg30min	5	30 minutes
bg60min	6	1 hour
bg120min	7	2 hours
bg300min	8	5 hours

### 6.10.8 BgQuantityType

To set quantity to update background

- Set through SetBackgroundQuantity(), GetBackgroundQuantity() method of Tof class.

Name	Value	Description
None	1	No update
bgLv1	2	Update Level1
bgLv2	3	Update Level2
bgLv3	4	Update Level3
bgLv4	5	Update Level4
bgLv5	6	Update Level5
bgLv6	7	Update Level6
bgLv7	8	Update Level7
bgLv8	9	Update Level8
bgLv9	10	Update Level9
Full	11	Full update

### 6.10.9 HumanStatusType

Status of a human

- status member of Human struct

Name	Value	Description
Untracked	-1	Not detected
Walk	0	Walking
Stand	1	Standing(&Stopping)
Crouch	2	Crouching
StandHand	3	Standing and reaching hand
CrouchHand	4	Crouching and reaching hand

### 6.10.10 CaptureStatus Type

To set status of capture function

- Set for GetCaptureStatus() method of Tof class.

Name	Value	Description
Disable	0	Disabled
Stop	1	Stopping
Run	2	Capturing

### 6.10.11 EdgeSignalCutoff Type

To set edge noise reduction mode

- Set through SetEdgeSignalCutoff() method of Tof class.

Name	Value	Description
Disable	0	Disabled
Enable	1	Enabled
Unknown	-1	Not set or Unknown Mode

## 6.10.12 Result Type

Return code of function (Error Code)

Name	Value	Description
OK	0	Success
SequenceError	-1	Sequence Error
ArgumentInvalid	-2	Argument Error
TimeOut	-3	Time-out Error
MemoryAllocError	-4	Memory Allocation Error
Unsupported	-5	Unsupported Error
SensorArgumentInvalid	-100	Sensor Argument Error
SensorConnectionFail	-101	Sensor Connection Error
SensorAlreadyConnected	-102	Sensor Already connected
SensorCommandFail	-103	Sensor Command Error
SensorTimeout	-104	Sensor Time-out Error
SensorOtherError	-105	Sensor Other Error
IniOperationFail	-200	ini file Related Error
SDKEnvPathFail	-201	SDK Environment Path error
DriverOpenFail	-202	TOF Driver Open error
TofConflictError	-203	TOF Conflict Error
OtherError	-1000	Other Error



## 7 Support Tools

### 7.1 Reference Code

#### 7.1.1 Tof2dViewer.cpp

##### 7.1.1.1 Outline

A sample program to display depth data received from TOF sensors in color by OpenCV. If multiple TOF sensors (Max 9 TOFs) are connected, the view is divided as multi displays (Refer to Figure 43).

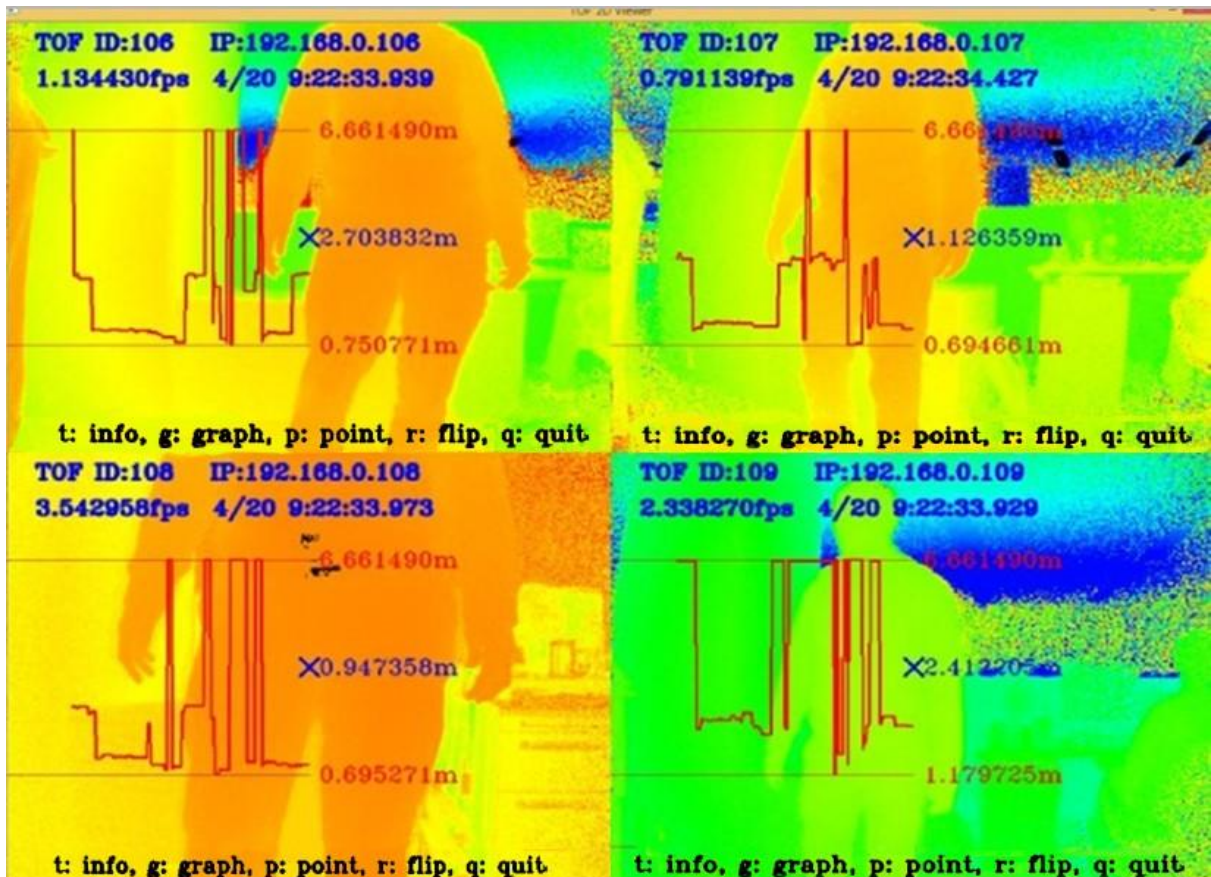


Figure 43 Display of Tof2dViewer.cpp

##### 7.1.1.2 Usage

After setting up TOF sensor as #4.5 and starting the application, display is started. Number of multiple displays depends on number of TOF sensors connected.

'q' : Quit the application.

't' : Toggle between display and hide TOF ID, FPS, timestamp.  
(Information is displayed, when the application is started.)

'g' : Display a graph of depth data chosen by the mouse.

'p' : Display the distance of the places chosen by the mouse.

'r' : Inverted and Displays.

## 7.1.2 Tof3dViewer\_cv.cpp

### 7.1.2.1 Outline

A sample program to convert depth data from TOF sensor to 3D and display on OpenCV. The image can be rotated, and it can be shown as top view for example (Refer to Figure 44).

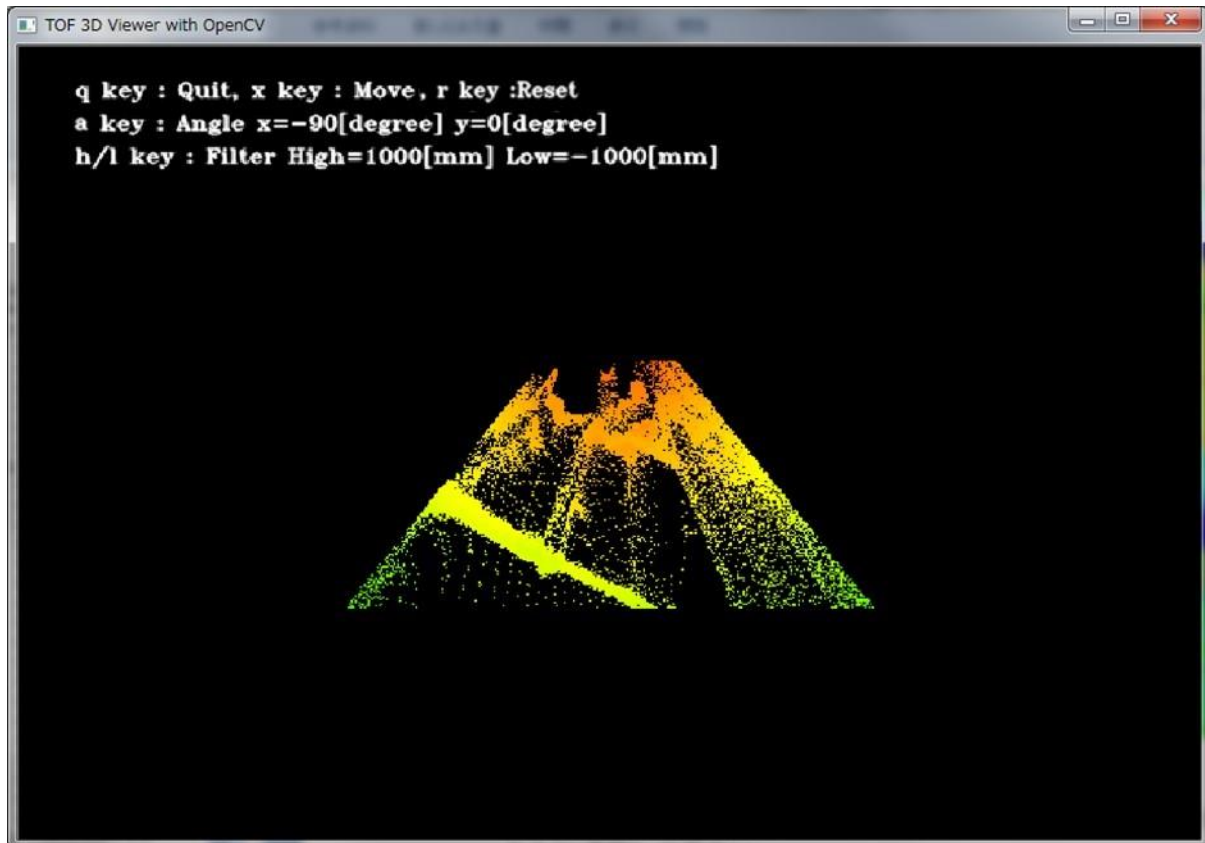


Figure 44 Display of Tof3dViewer\_cv.cpp

### 7.1.2.2 Usage

After setting up TOF sensor as #4.5 and starting the application, display is started.

'q' : Quit the application.

'a' : Rotate the image. (Default rotate angle is 0 degrees)

Up/down key: Rotate the image about the X-axis.

Left/right key: Rotate the image about the Y-axis.

'x' : Move the image. (Default position is center)

Up/down key: Move the image to Y-coordinate.

Left/right key: Move the image to X-coordinate.

'h' : Up/down key changes the level of highest display, (Default level is 1000mm)

'l' : Up/down key changes the level of lowest display. (Default level is -1000mm)

'r' : Restore to default settings.

### 7.1.3 Tof3dViewer\_pcl.cpp

#### 7.1.3.1 Outline

A sample program to convert depth data from TOF sensor to 3D and display on PCL (Point Cloud Library). The 3D figure can be moved freely by mouse (Refer to Figure 45).

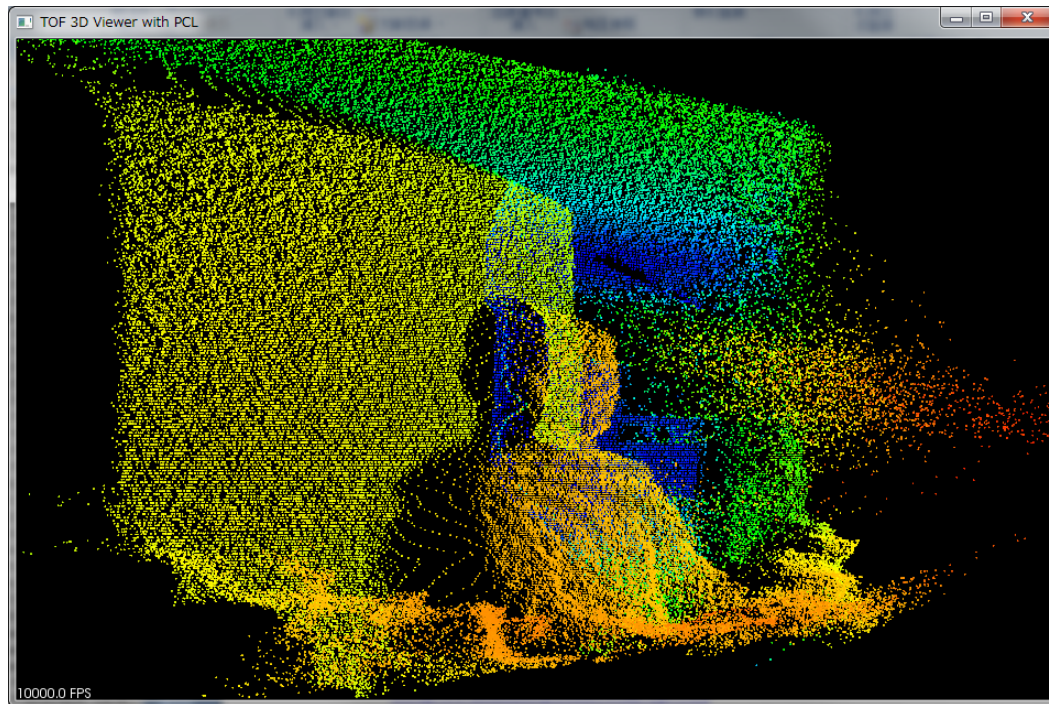


Figure 45 Tof3dViewer\_pcl.cpp

#### 7.1.3.2 Execute method

- 1) Install this SDK as #4.4.
- 2) Install PCL (Point Cloud Library).  
PCL 1.8.0 can be downloaded from the following site.  
<http://unanancyowen.com/pcl18/>  
\* If old version of PCL such as PCL 1.7.2 is already existing in the environment, it must be uninstalled.
- 3) Set system environment.

Name	Value
PCL_ROOT	C:/Program Files/PCL 1.8.0 (or C:/Program Files(x86)/PCL 1.8.0)
Path	;%PCL_ROOT%\bin for x86, ; %OPENNI2_REDIST% for x64, ;%OPENNI2_REDIST64%

#### 7.1.3.3 Usage

After setting up TOF sensor as #4.5 and starting the application, display is started.

'q' : Quit the application.

The figure can be rotated and zoom in and out by mouse operation.

'1' : Enable or disable noise filter.

'2' : Change resolution to VGA or QVGA. (Default of the application is QVGA.)

'3' : Switch the background to white and black.(Default is black)

### 7.1.4 TofIrViewer.cpp

#### 7.1.4.1 Outline

A sample program to display IR data received from TOF sensors in monochrome picture by OpenCV. If multiple TOF sensor (Max 9 TOFs) are connected, the view is divided as multi displays (Refer to Figure 46).



Figure 46 Display of TofIrViewer.cpp

#### 7.1.4.2 Usage

After setting up TOF sensor as #4.54.5 and starting the application, display is started. Number of multiple displays depends on number of TOF sensors connected. The IR gain (the brightness) can be controlled by the level of 1-15 by the upper and lower key (When more than one TOF sensor is connected, all TOF sensors will be the same setting.) Push 'q' key to quit the application.

## 7.1.5 HumanCounter.cpp

### 7.1.5.1 Outline

This is a sample application to count humans using human detection feature of the SDK who enters into and exits from indicated area.

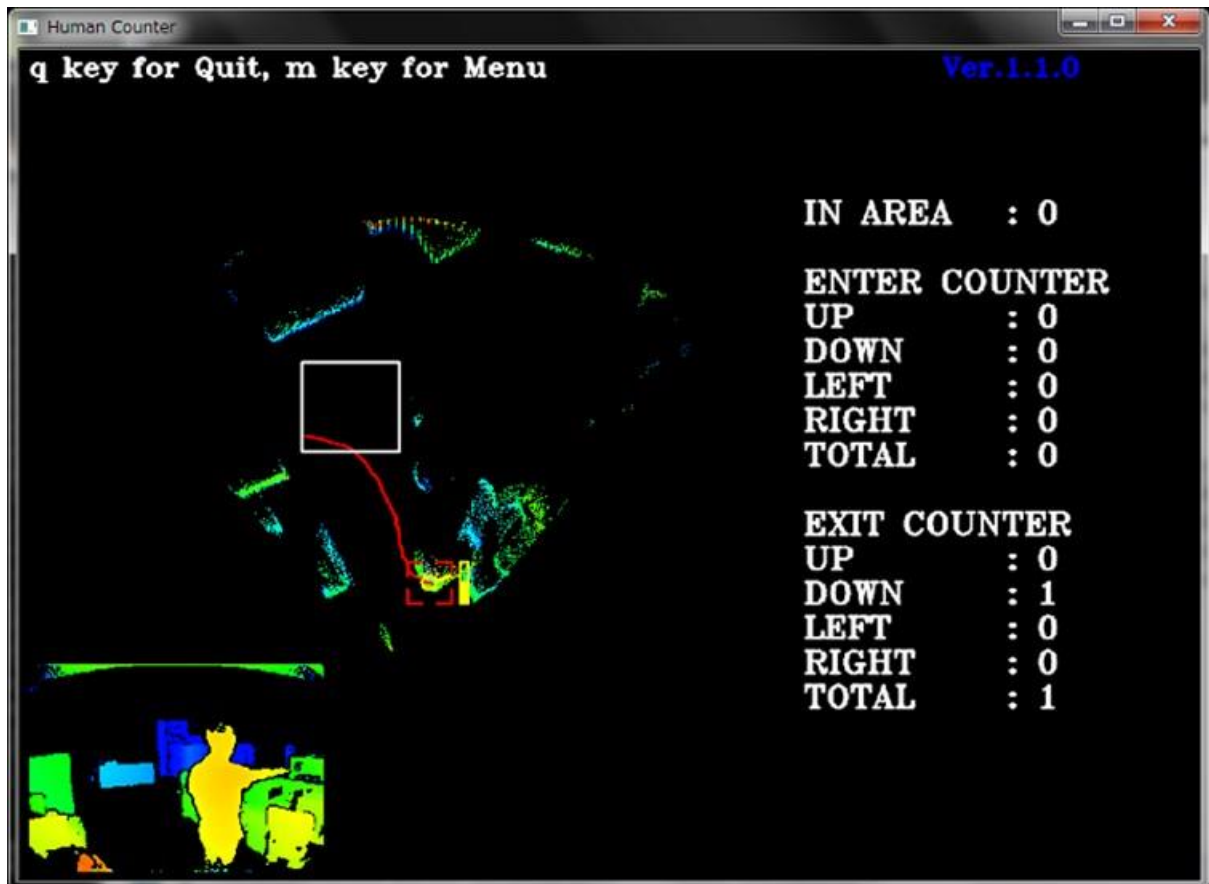


Figure 47 HumanCounter.cpp



### 7.1.5.2 Setup

- 1) Setup and turn on TOF sensor as #4.5.
- 2) Edit IP address in tof.ini.
- 3) Execute HumanCounter.exe.
- 4) Push 'a' key to set X-axis rotation and Z-axis rotation.

Adjust X-axis rotation by Up/down key till TOF sensor angle to be flat from 'Side View'.

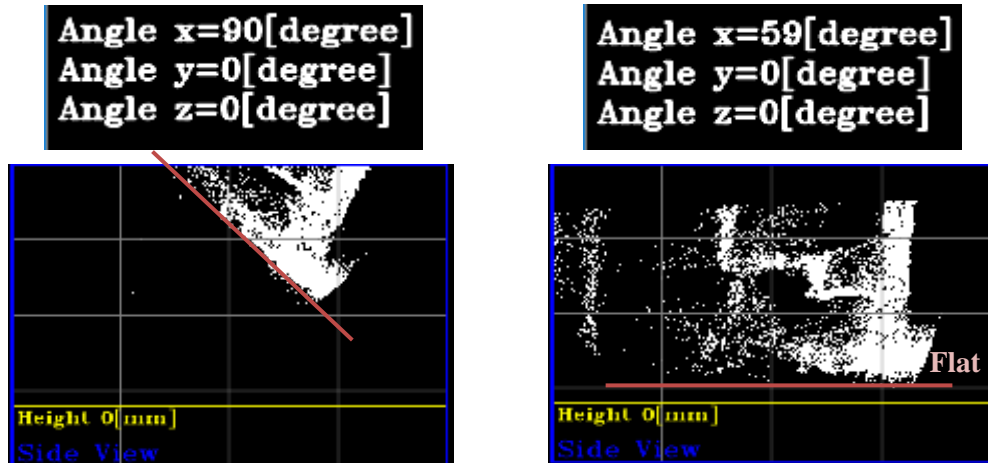


Figure 48 Adjusting TOF sensor angle (X-axis)

Adjust Z-axis rotation by Left/Right key till TOF sensor angle to be flat from 'Front View'.

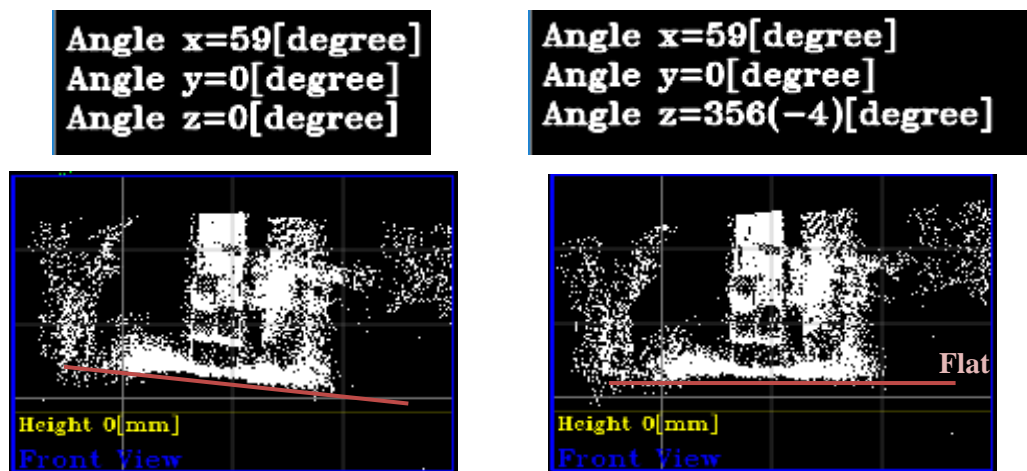


Figure 49 Adjusting TOF sensor angle (Z-axis)

- 5) Push 'h' key and adjust height position of TOF sensor by Up/down key to match the floor line on 'Front View' or 'Side View'.

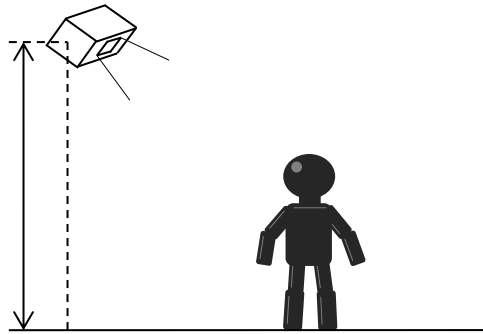


Figure 50 Set height position of TOF sensor

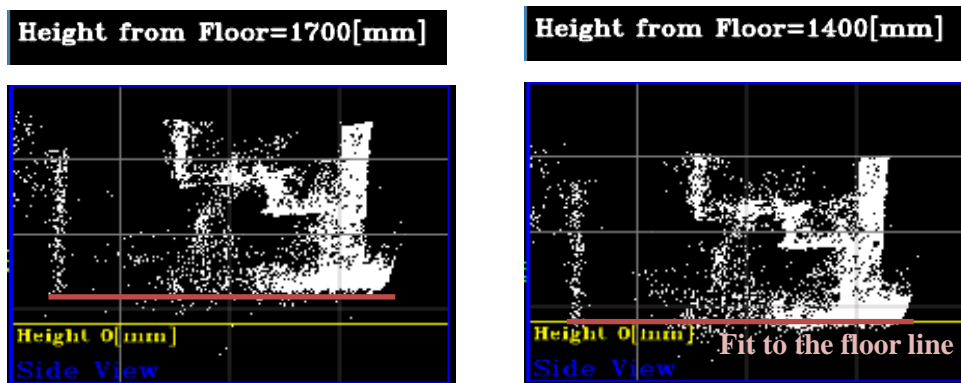


Figure 51 'Side View' to adjusting height position of TOF sensor

- 6) Push 's' key to shift the screen by up / down / left / right keys.
- 7) Push 'z' key to zoom in or zoom out the screen by up / down keys.
- 8) Push 'b' key to setup human count area if human counting is used.
  - If "Position" is selected, human count area can be shifted by up / down / left / right keys.
  - If "Size" is selected, lower right point can be moved by up / down / left / right keys.
  - "Position" and "Size" are switched by 'b' key.

Setting values are saved to HumanCounter.ini file in the same folder with the application when the application is terminated.

### 7.1.5.3 Usage

'q' : Quit the application after pushing 'y' key and saving settings to HumanCounter.ini.

'm' : Display menu.

'p' : Set display settings pushing the following numbers.

'1' : Points ON/OFF (Default ON)

'2' : Footprints ON/OFF (Default OFF)

'3' : Human Counter ON/OFF (Default ON)

'4' : Subdisplay (Sensor view) ON/OFF (Default ON)

'9' : Reset Footprints (available only if Footprints is ON.)

'0' : Reset Human Counter (available only if Human Counter is ON.)

'e' : Change detection area.

'Position' : Move count area by up/down/left/right keys.

'Size' : Resize the count area by up/down/left/right keys.

'Position' and 'Size' are toggled with the 'e' key.

'f' : Save the screen to [date/time].png.



## 7.1.6 Tofv2Viewer.cpp

TOF	TOFv2 or later
SDK	v2.0.0 or later

### 7.1.6.1 Outline

A sample program for TOFv2 to display depth and IR simultaneously, and background subtraction. Depth, IR, background and motion are displayed to main display and sub display (Refer to Figure 52). This application cannot be used for TOFv1 sensor.



Figure 52 Display of Tofv2Viewer.cpp

### 7.1.6.2 Usage

After setting up TOF sensor as #4.5 and starting the application, display is started.

- 'q' : Quit the application after pushing 'y' key.
- 'm' : Display menu.
- 'p' : Set display settings pushing the following numbers.
  - '1' : Sub Display ON/OFF (Default is ON)
  - '2' : Change Sub Display position (Default is lower-left)
  - '3' : Switch Sub Display and Main Display
  - '4' : Change displayed contents (Change CameraMode) (Default is Depth and IR)
- 'b' : Change background subtraction settings.
  - Up/Down : Change background update interval (Default is 1 minute)
  - Left/Right : Change background update quantity (Default is Level1)
- 'r' : Reset background
- 'o','i' : Start capturing and replaying.
- 't' : Change text color.
- 'f' : Save screen.

## **7.2 Firmware Updater**

TBD

## 8 Terminologies

- ☑ **API (Application Programming Interface)**  
Specifies how some software components should interact with each other. In practice, most often an API is a library that includes specifications for routines, data structures, object classes, and variables. An API specification can take many forms, including an International Standard such as POSIX, vendor documentation such as the Microsoft Windows API, the libraries of a programming language, e.g., Standard Template Library in C++ or Java API.
- ☑ **CGI (Common Gateway Interface)**  
Execute programs on web server for many types of request from clients.
- ☑ **FPGA (Field Programmable Gate Array)**  
An integrated circuit that any logic can be implemented by customers.
- ☑ **HLDS (Hitachi LG Data Storage, Inc.)**  
A joint venture between Hitachi and LG. The company is an OEM manufacturer of optical data storage devices such as CD and DVD drives for desktop and notebook computers and it has been Global No.1 share over 10 years.
- ☑ **IR (Infrared)**  
Invisible light which wavelength is longer than visible red light. It is electromagnetic radiation which wavelength is approximately 0.7um to 1mm. Hitachi-LG Data Storage's TOF sensor uses 0.85um Laser Diode.
- ☑ **PoE (Power over Ethernet)**  
A technology to supply electrical power to devices through UTP (Unshielded Twist Pair) Ethernet cable. PoE+ (IEEE802.3a) can supply 25.5W through a port, it can cover a TOF sensor.
- ☑ **RTP (Real-time Transport Protocol)**  
A network protocol on Application layer to transfer data stream of audio and video in real-time. UDP (User Datagram Protocol) is used on Transport layer.
- ☑ **SDK (Software Development Kit)**  
A package of software, hardware, debug tools, documents and so on for application developers who uses the software product or the hardware product.
- ☑ **TOF (Time-of-Flight)**  
Refer to Chapter 2.1.
- ☑ **UTC (Universal Time, Coordinated)**  
The primary time standard by which the world regulates clocks and time.
- ☑ **XML (Extensible Markup Language)**  
A markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.