對外秘

Time-of-Flight Sensor

Calibration Guide

Revision 2.0
June 10, 2019

SAMSUNG ELECTRONICS

Beyond You Can Imagine

Important Notice

Samsung Electronics Co., Ltd.
San #24 Nongseo-Dong, Giheung-Gu
Yongin-City, Gyeonggi-Do, Korea 446-711

Contact Us: *kh74.bae@samsung.com*

Home Page: *http://www.samsungsemi.com*

Revision History

Revision No.	Date	Detailed of Modification	Drafter
1.0	Mar. 30, 2019	Initial Calibration Guide	K. Bae, H. Ko, D. Seo, M. Kang, C. Lee, M. Bae, S. Son, M. Keel
2.0	Jun. 10, 2019	Changed the calibration methods 1) Temperature drift calibration independent on wiggling and FPPN 2) Wiggling calibration uses a checkerboard chart which is the same equipment in Lens-FPPN. Removed the time-dely board part which is mentioned in a separate document.	K. Bae, H. Ko, D. Seo, C. Lee

Contents

1 OVERVIEW 6
1.1 Systematic Errors 6
1.1.1 Wiggling 6
1.1.2 Fixed Phase Pattern Noise (FPPN) 6
1.1.3 Temperature Drift 6
1.1.4 Lens 7
1.2 Overall Calibration Procedure 7
2 CALIBRATION 11
2.1 Temperature Drift 11
2.1.1 Procedure 11
2.1.2 Equipment 12
2.2 Wiggling, Lens-FPPN 14
2.2.1 Procedure 15
2.2.2 Equipment 16
2.3 Lens-FPPN 18
2.3.1 Procedure 19
2.3.2 Equipment 22
3 SOFTWARE 24
3.1 Overview 24
3.2 Temperature Drift Tool 24
3.2.1 Usage 25
3.2.2 Deliverables 28
3.2.3 Temperature Drift Monitor Configuration file 29
3.2.4 Temperature Drift Monitor DLL API 29
3.2.5 Temperature Drift Monitor Output 35
3.3 Calibration Library 35
3.3.1 Deliverables 35
3.3.2 API 36
3.3.3 Calibration Configuration file 36
3.3.4 Calibration Result Output 39
3.4 DEPS Tool 39
REFERENCES 40

List of Figures

Figure Number | **Title** | **Page Number**

Figure 1.1 Systematic Error Sources 6
Figure 1.2 Overall Calibration Procedure 7
Figure 1.3 M2M Calibration Procedure 8
Figure 1.4 Overall Compensation Procedure 9
Figure 1.5 Example of Overall Compensation Procedure 10
Figure 2.1 Temperature Drift Calibration Procedure 12
Figure 2.2 Example of temperature drift compensation equipment using a plane chart. Adjustable height stage mount for module (400~700mm) which is combined with TEC. Height will be fixed for a manufacturing environment. 13
Figure 2.3 Temperature Calibration Board 13
Figure 2.4 Comparison between a moving chart ((a) and (b)), and a virtual chart by time-delay ((c) and (d)) 15
Figure 2.5 Wiggling Calibration Procedure 16
Figure 2.6 Wiggling and Lens-FPPN Calibration Chart. Height adjustable stage mount for module (375 – 825 mm) and an angle adjustable chart (20º – 40º). Height and angle will be fixed for a manufacturing environment. 17
Figure 2.7 The Wiggling Calibration Board. 18
Figure 2.8 Lens-FPPN Calibration Procedure 20
Figure 2.9 Lens Calibration Procedure. 20
Figure 2.10 FPPN Calibration Procedure; each rotation. 21
Figure 2.11 FPPN Calibration Procedure; merge all rotations. 22
Figure 2.12 Lens-FPPN Calibration Chart. Height adjustable stage mount for module (375 – 825 mm) and an angle adjustable chart (20º – 40º). Height and angle will be fixed for a manufacturing environment. 23
Figure 3.1 Released Softwares; 1) Temperature Drift Tool, 2) Calibration Library, 3) Validation Tool. 24
Figure 3.2 Temperature Drift Calibration Block Diagram 25
Figure 3.3 Temperature Drift Monitor GUI 25
Figure 3.4 Temperature Drift Monitor Options 26
Figure 3.5 Set Path Dialog 26
Figure 3.6 Temperature drift offset from reference depth caused by self-induced heating 27
Figure 3.7 Display Depth Map 28
Figure 3.8 Display Information 28
Figure 3.9 Temperature Drift Monitor Configuration XML example 29
Figure 3.10 Temperature Drift Monitor API Guide 32
Figure 3.11 Calibration Configuration XML example 37

對外秘

1 Overview

A system calibration is a process to reduce the effect of the systematic errors in a ToF system, as shown in Figure 1.1. This document aims to present calibration method, equipment, and software.

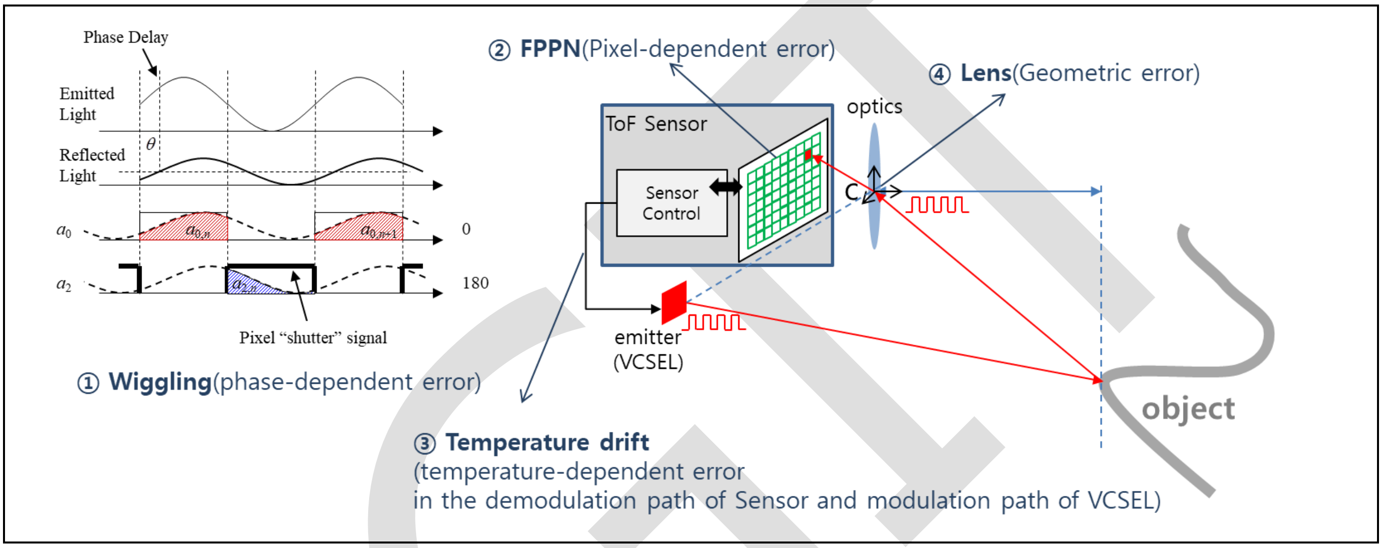

Figure 1.1 Systematic Error Sources

1.1 Systematic Errors

1.1.1 Wiggling

Phase (distance)-dependent error due to harmonic distortion.

1.1.2 Fixed Phase Pattern Noise (FPPN)

Pixel-dependent error due to the time-delay of demodulation signal depending on individual pixel position and the misalignment of a VCSEL and a sensor (which is geometric error).

1.1.3 Temperature Drift

Temperature-dependent error because the time-delay with respect to temperature can be changed in the demodulation path of a sensor and in the modulation path of a VCSEL.

1.1.4 **Lens**

Geometric error due to lens distortion, shifts, and tilts.

1.2 **Overall Calibration Procedure**

There are two types of ToF calibration

- One-time calibration: temperature drift calibration is typically performed once. All modules can use the same parameters.
- Module to Module (M2M) calibration: wiggling, lens, and FPPN have per-module variation. Each module must be calibrated.

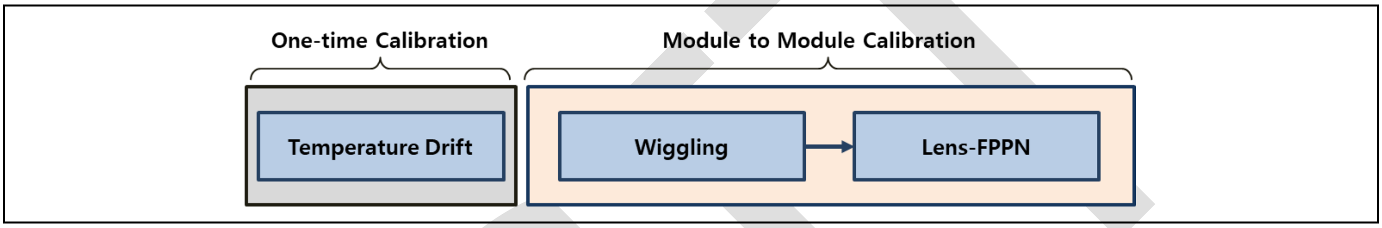

Figure 1.2 Overall Calibration Procedure

<u>One-time calibration</u>

1) Obtain the reference depth using the Temperature Drift Tool.
2) Obtain the statistics of temperature drift depending on the time and ambient temperature using the Temperature Drift Tool.
3) Calculate the coefficients using the statistics.
4) Two outputs; ① data for depth ISP ② data for M2M calibration.

<u>M2M Calibration</u>

: During the M2M calibration, the temperature in a sensor and a VCSEL driver will change over time. Temperature drift compensation is performed at each calibration.

1) Calibrate wiggling **<u>with the external time-delay.</u>**
2) Calibrate lens-FPPN **<u>without the external time-delay (normal mode).</u>**

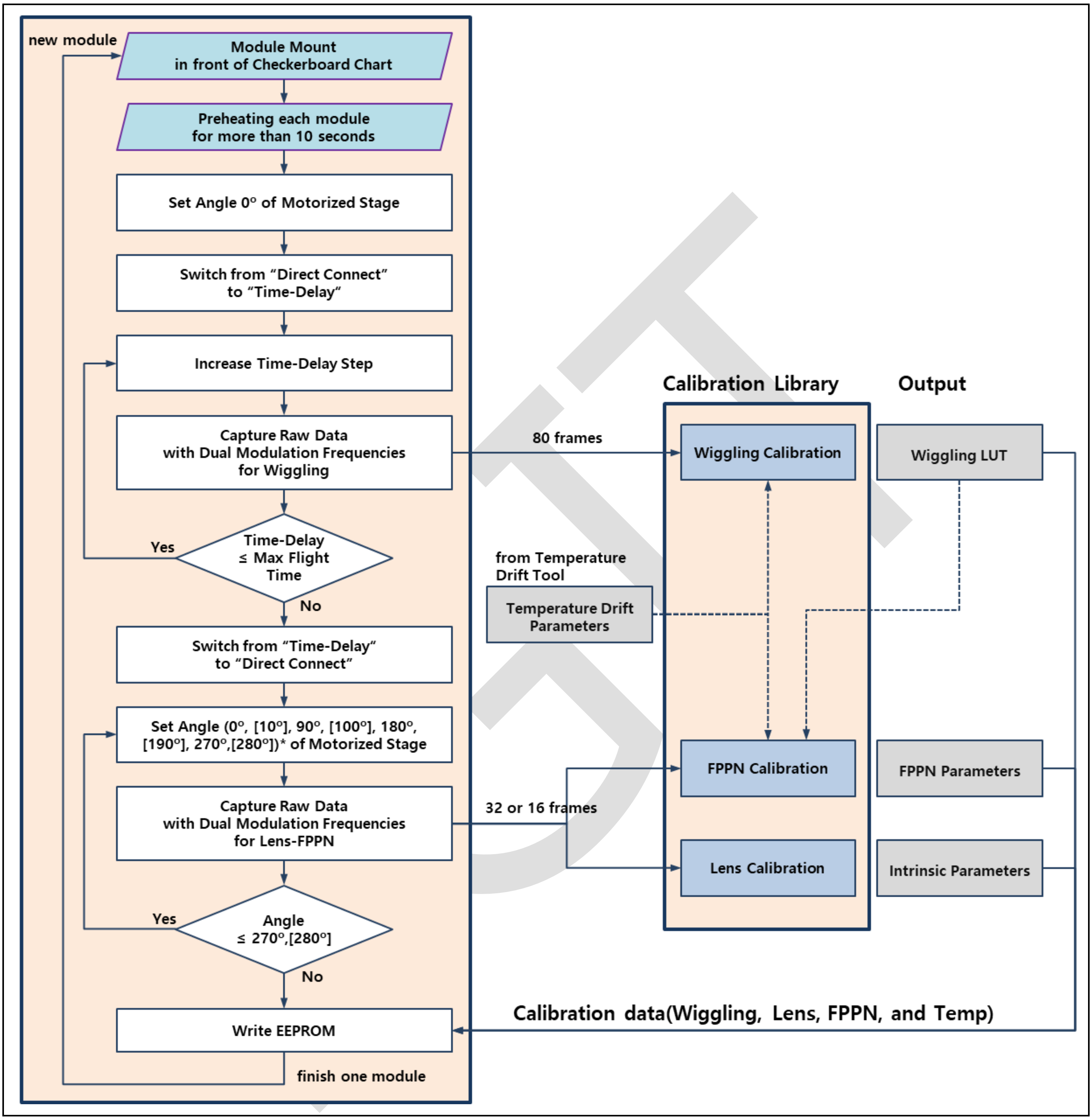

Figure 1.3 M2M Calibration Procedure

對外秘

Table 1.1 M2M Calibration Processing Time (conditions: dual modulation frequencies; 100MHz & 80MHz, PC Spec.; i7 3.6GHz, RAM 16GB, frame grabber; Simmian MV3 board).

Calibration items	preheating time (sec)[1)]	data capturing time (sec)	data loading and processing time (sec)	Num of data (depth/frame)[5)]
wiggling	10	3.5 [2)]	1.5	20/80
lens-FPPN	-	40 [3)]	7.8 [4)]	8/32
total	10	43.5	9.3	28/112
	62.8			

1) Preheating time can be reduced depending on the temperature drift performance.
2) Wiggling data capturing time measured with Samsung's Wiggling Calibration Board and Simmian board.
3) 8 different views of a plane chart (0, 10, 90, 100, 180, 190, 270, 280°) are assumed. Lens-FPPN data capturing time depends on the speed of rotation and stabilization of motorized stage. If 4 different views of a plane chart are used the number of data have to be increased after evaluating a module.
4) If 4 different views of a plane chart are used, the data loading and processing time is reduced up to 5s.
5) One depth requires 4 raw frames.

Figure 1.4 shows the compensation procedure using calibration data. Figure 1.5 represents the example of overall compensation procedure.

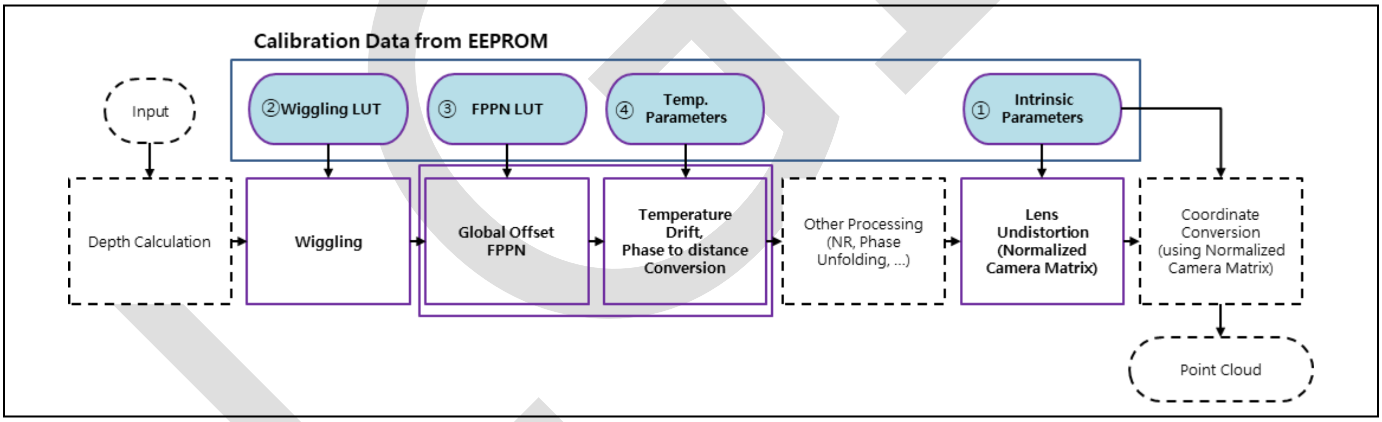

Figure 1.4 Overall Compensation Procedure

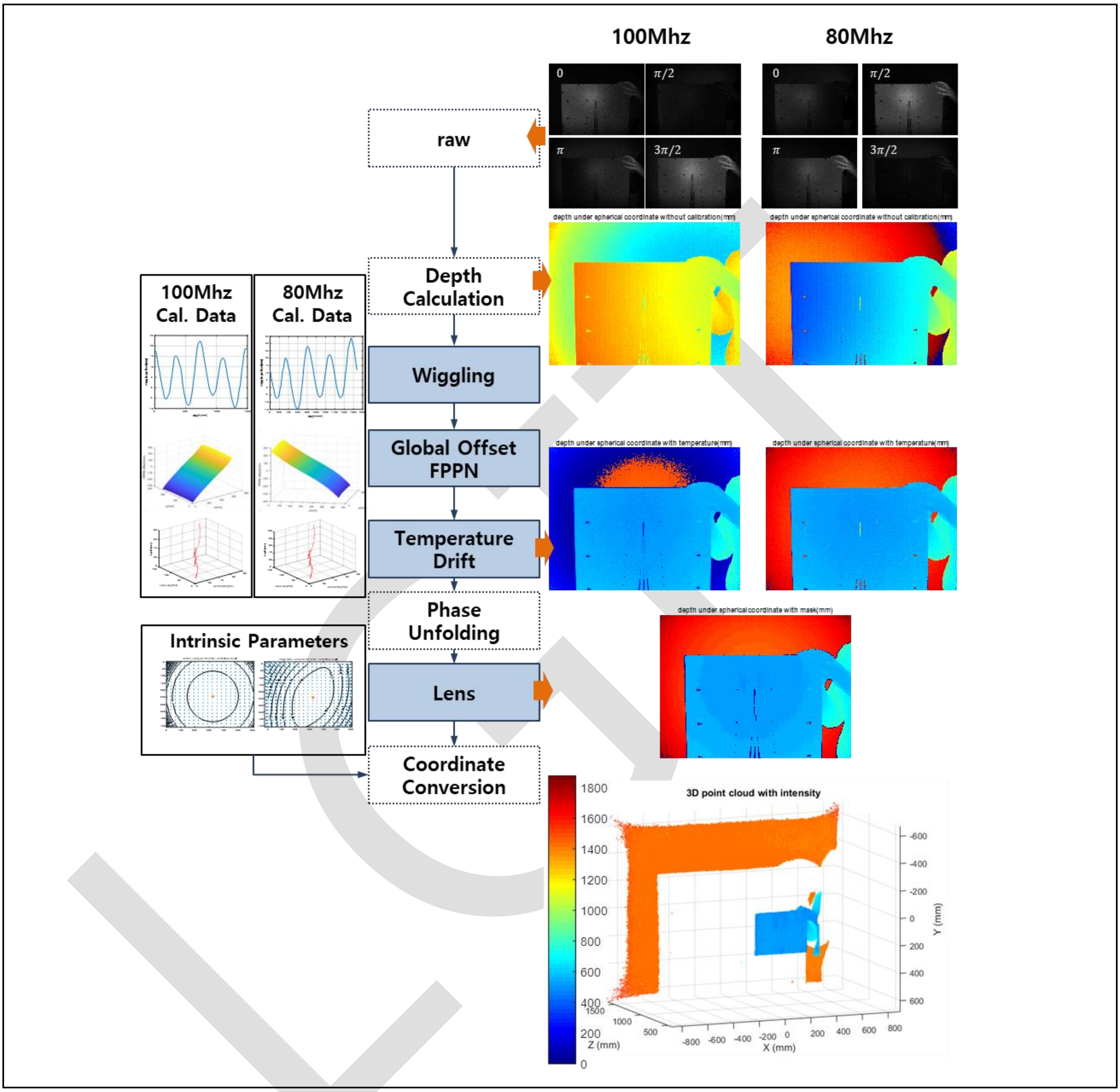

Figure 1.5 Example of Overall Compensation Procedure

2 Calibration

2.1 Temperature Drift

Temperature can change the depth offset, because the delay time in the VCSEL modulation and sensor demodulation path can be altered with temperature. Therefore, the distance offset compensation with temperature drift should be applied to the original distance, based on the temperatures in a sensor and a laser driver (here, we assume temperature of the VCSEL is represented by a laser driver's temperature). Compensation offset can be modeled as follows:

$$compensation\ offset = f(x, y)$$

where $x = temp_{sensor}$ and $y = temp_{vcsel}$ are the temperature data of a sensor and a laser driver, respectively, which can be obtained in the embedded line of sensor raw data (see more detailed information from the application note of the ToF sensor). $f(x, y)$ is a non-linear function. The coefficients of a function are calculated using temperature drift tool with respect to the temperature data. Temperature drift calibration is typically performed only one time from mass data, but not in M2M.

Temperature drift statistics can be obtained by "Temperature Drift Tool" depending on time and ambient temperature. Ambient temperature may be changed while obtaining statistics. "Temperature Drift Tool" calculates the coefficients from these statistics.

2.1.1 Procedure

Temperature Drift Calibration Procedure (see Figure 2.1)

1) Select a camera module for temperature drift calibration or golden samples.

2) Set ambient temperature as 25℃

   a) Obtain the temperature statistics for 20 minutes at 25℃.

   : "Temperature Drift Tool" can obtain the reference depth.

3) Set ambient temperature as 5, 15, 25, 35, 45 and 55℃

   a) Obtain the temperature statistics for 10 minutes for heating upto each ambient temperature.

   b) Obtain the temperature statistics for 10 minutes for cooling.

   : "Temperature Drift Tool" can obtain the statistics of temperature drift.

3) Calculate coefficients using "Temperature Drift Tool".
   Obtain two sets; one set for EEPROM data and one set for M2M calibration. EEPROM data will be written on the all camera modules.

※ Note that if there is no temperature drift file for M2M calibration we provide the default file for M2M calibration. But we don't recommend it. It is used for only testing mode.

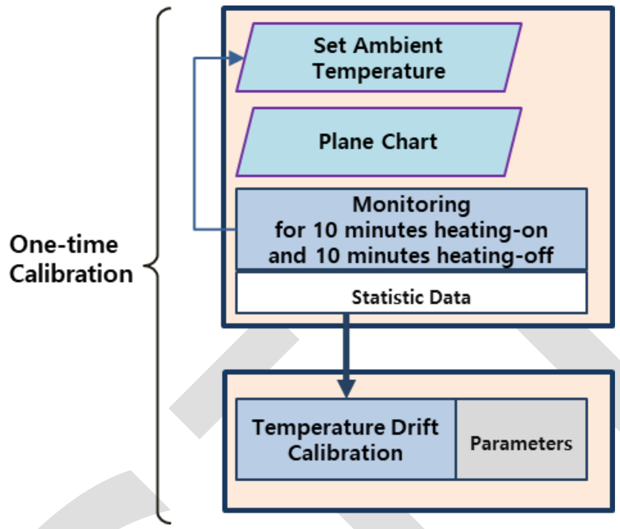

Figure 2.1 Temperature Drift Calibration Procedure

2.1.2 **Equipment**

For the temperature drift calibration, a plane chart whose reflectivity is higher than 80% is used in common. The distance from a module and a plane chart is fixed by monitoring the code to prevent saturation. This distance would be about 400 – 700 mm depending optical power of the VCSEL, integration time, and chart reflectivity.

Temperature drift calibration requires a sensor and a laser driver to be held at a specific stable temperature. For this purpose, a thermal chamber may be a candidate for adjusting temperature. However, size limitation of the thermal chamber can cause some problems. A small-size chamber will cause multiple reflections because a plane chart cannot be allowed to put into the chamber, and large-size chamber is costly and occupies huge space. If you have a large-size chamber you can use it.

We recommend a thermoelectric cooler (TEC) to regulate the calibration temperature. A TEC is a solid state device which can control heat flux using electric current; it is composed of a precision temperature controller and a base plate, as shown in Figure 2.2. Highly thermal conductive materials such as copper or aluminum will be used as the base plate for the chuck or thermal platform. And the TEC chip is directly connected to the base plate to control temperature fast and in bi-directional way. This allows for temperature settings above and below the ambient temperature. The TEC is particularly useful in small scale temperature control, providing fast temperature response and ultra-high temperature stability. Under low temperature test, it is important to create moisture free environment [1].

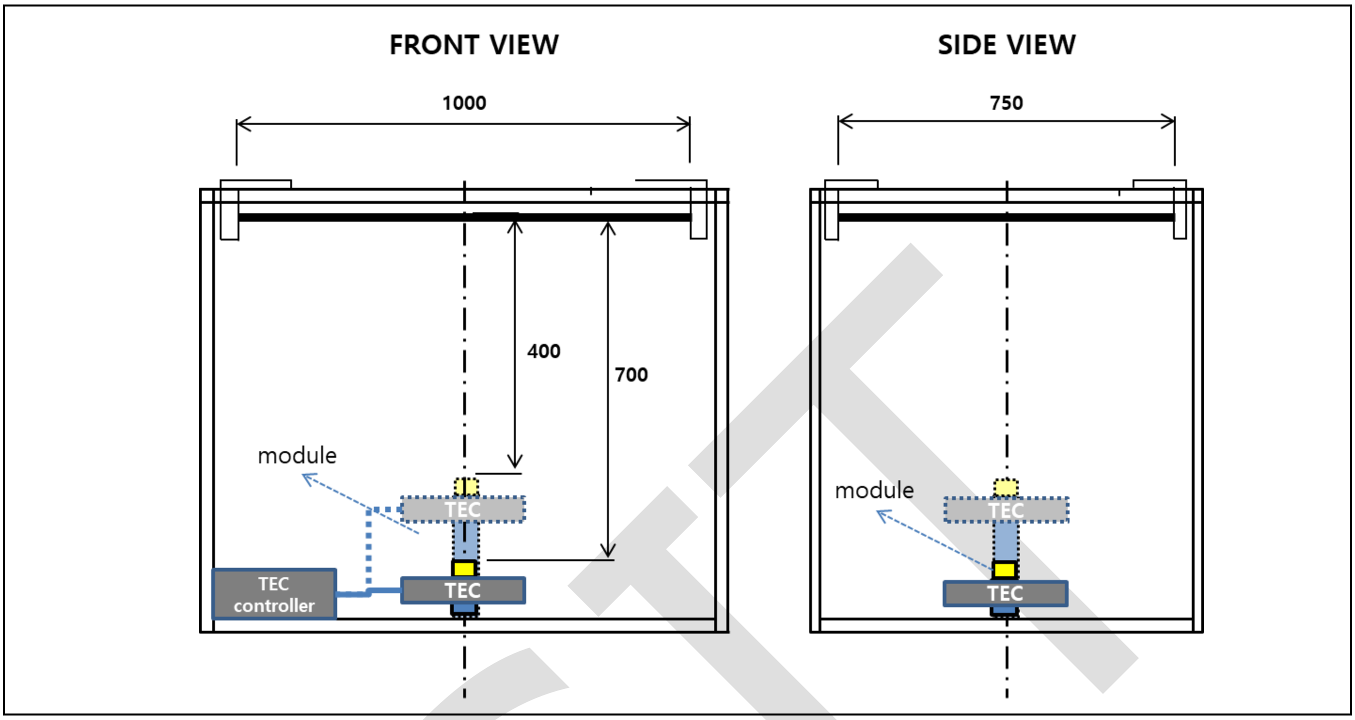

Figure 2.2 Example of temperature drift compensation equipment using a plane chart. Adjustable height stage mount for module (400~700mm) which is combined with TEC. Height will be fixed for a manufacturing environment.

When calibrating temperature drift, “Lens-FPPN & Temperature Drift Calibration Board” without the external time-delay as shown in Figure 2.3 has to be used.

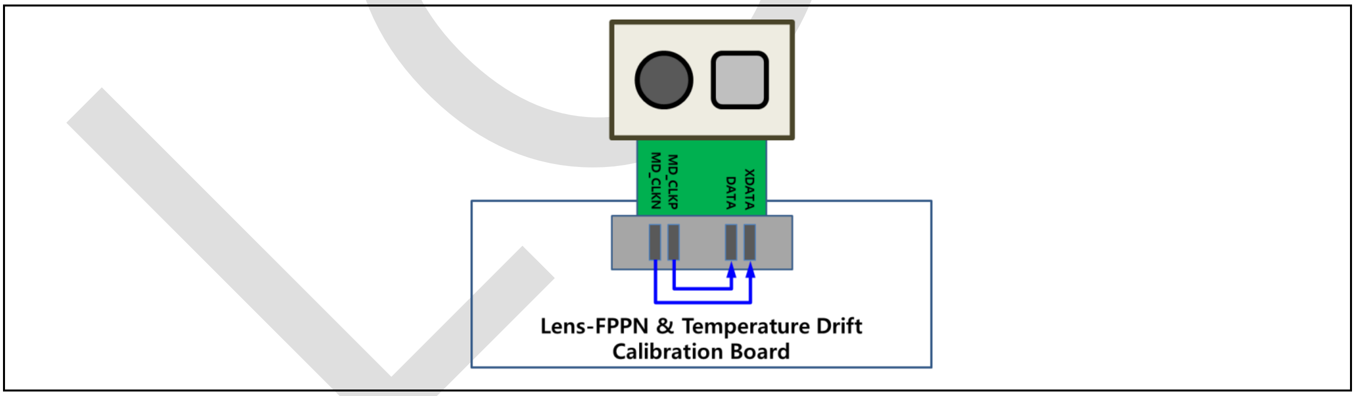

Figure 2.3 Temperature Calibration Board

2.2 **Wiggling, Lens-FPPN**

The wiggling error, also called as circular error, is a known systematic sinusoidal depth distortion. The error varies with the distance. Note that since the wiggling error is a nonlinear function of distance, in general, a look up table (LUT) is used; the residual error between the measured depth and the ground truth (GT) will be pre-calculated in the calibration procedure. In the conventional approach, the wiggling calibration is performed by a moving chart on a measurement track line. But in the mass production, this method cannot be used because of the space and capture time limitation, even though only the center region of interest (ROI) is used.

We will use an external time-delay board and a checkerboard chart to calibrate wiggling error. A checkerboard chart is the same as a chart for lens-FPPN. The procedure does not require moving chart on a measurement track line. Only a checkerboard chart in a fixed distance is sufficient. This method reduces calibration time and space for the mass production. It is possible to use only small data set. The time delay in the modulation signal for the VCSEL driver should be adjusted by an external time-delay board in order to add a phase shift between the modulation and the demodulation of the light signal. Delay in the modulation path is equivalent to a distance shift of the object at an optical center.

Note that the raw data of the charts obtained by an external time-delay board will be different from those obtained by a moving chart as shown in Figure 2.4. In case of the moving chart, the distance from the sensor and the chart increases gradually in z-direction on the Cartesian coordinates system of the sensor (see Figure 2.4 (a) and (b)). However, when using the time-delay circuit, the virtual distance from the sensor and the chart is consistently adjusted in r-direction on the Spherical coordinates system, but not in z-direction on the Cartesian coordinates system. Therefore, the resultant raw data looks like capturing curved surfaces as shown as in Figure 2.4 (b) and (d).

For the test of this method, the precision time-delay ICs such as NB6L295 from On-semiconductor® are used. The NB6L295 is a dual channel programmable delay chip [2]; the main specifications are 0 – 11.2 ns total time delay (extended mode), 11ps time delay per step, 511 step (9-bit control), linearity of +/-20ps (max.). The delay time can be digitally controlled by an MCU board such as Arduino. A single time step of the NB6L295 is equivalent to about 1.54 mm distance shift. Since the time-delay IC contains 2048 steps when used by two chips in series, the maximum delay represents a distance shift of about 3.1478m.

對外秘

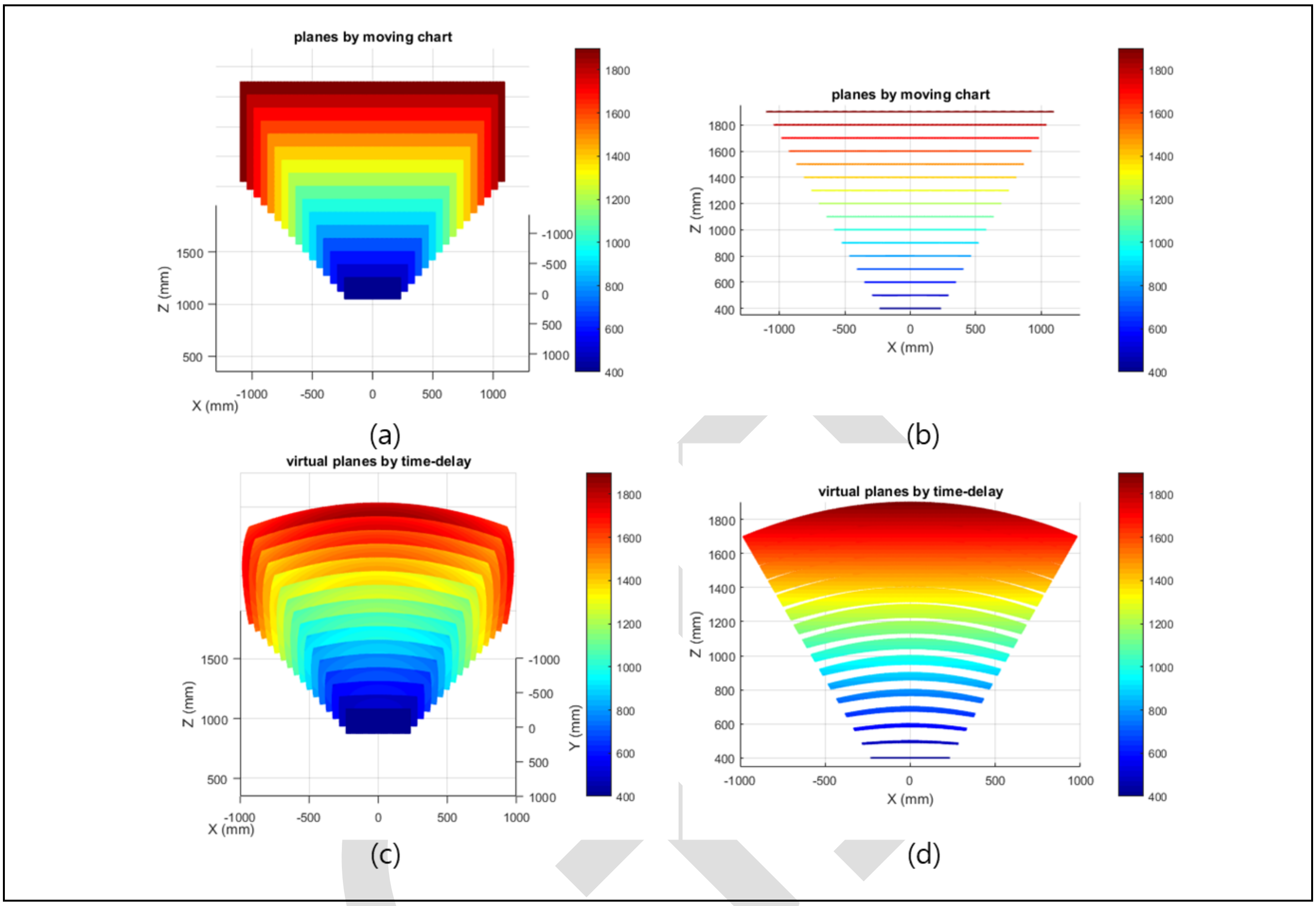

Figure 2.4 Comparison between a moving chart ((a) and (b)), and a virtual chart by time-delay ((c) and (d))

2.2.1 **Procedure**

Wiggling Calibration Procedure (see Figure 2.5)

1) Switch from “Direct Connect” to “Time-Delay” using Jumper or Relay Switch

2) Set a module to be able to capture a checkerboard chart

3) Preheat the module for more than 10 seconds.

4) Set the number of the steps to specify the number of images used in calibration (default, 20)

5) Raw data is captured automatically.

6) Calculate wiggling LUT using Calibration Library.

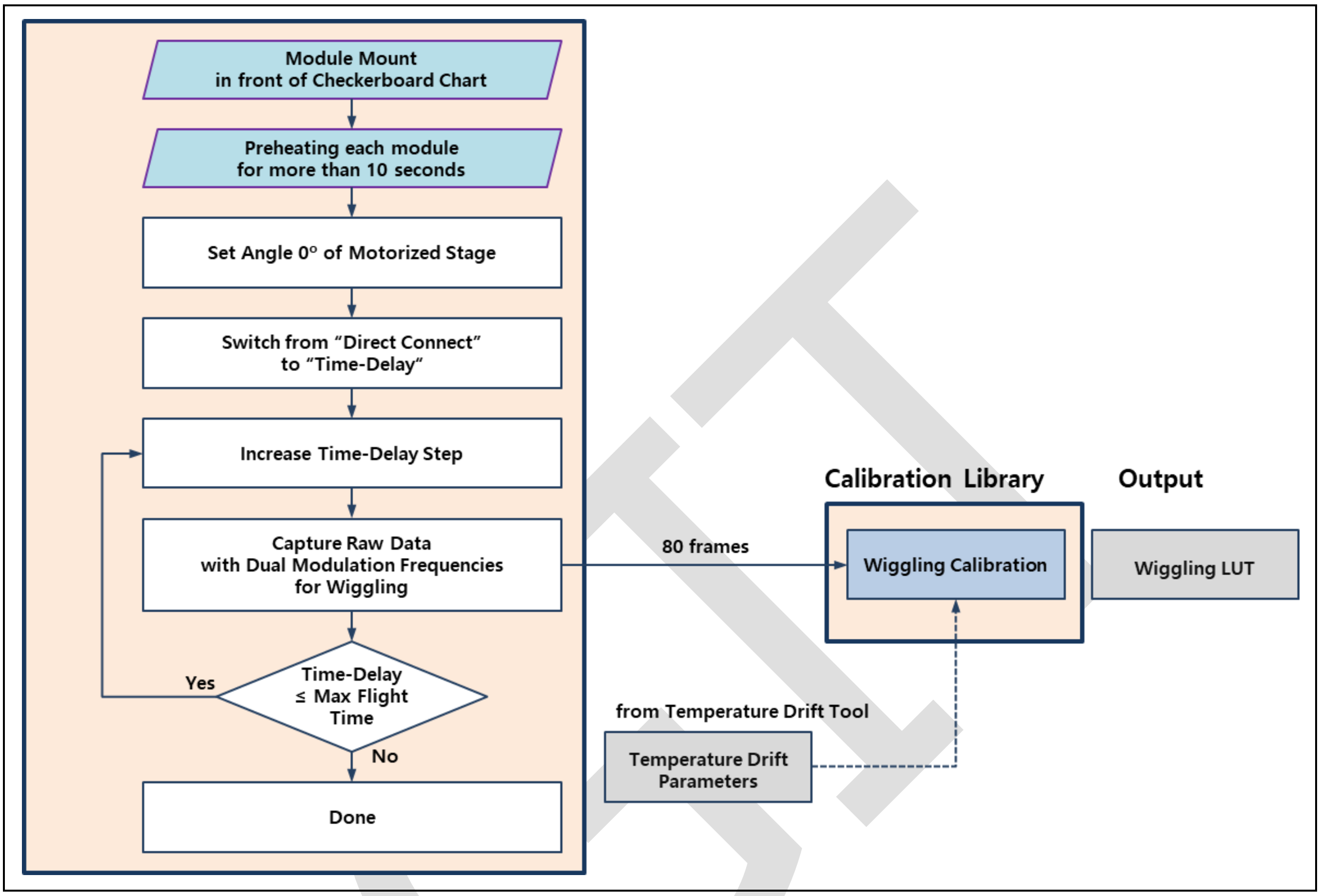

Figure 2.5 Wiggling Calibration Procedure

2.2.2 **Equipment**

For the wiggling calibration, a checkerboard chart can be used same as lens-FPPN. The reflectivity of white checker is more than 80%. The distance from a module and a plane chart is fixed by monitoring the code to prevent saturation. This distance is about 400 – 700 mm depending optical power of VCSEL, integration time, and chart reflectivity.

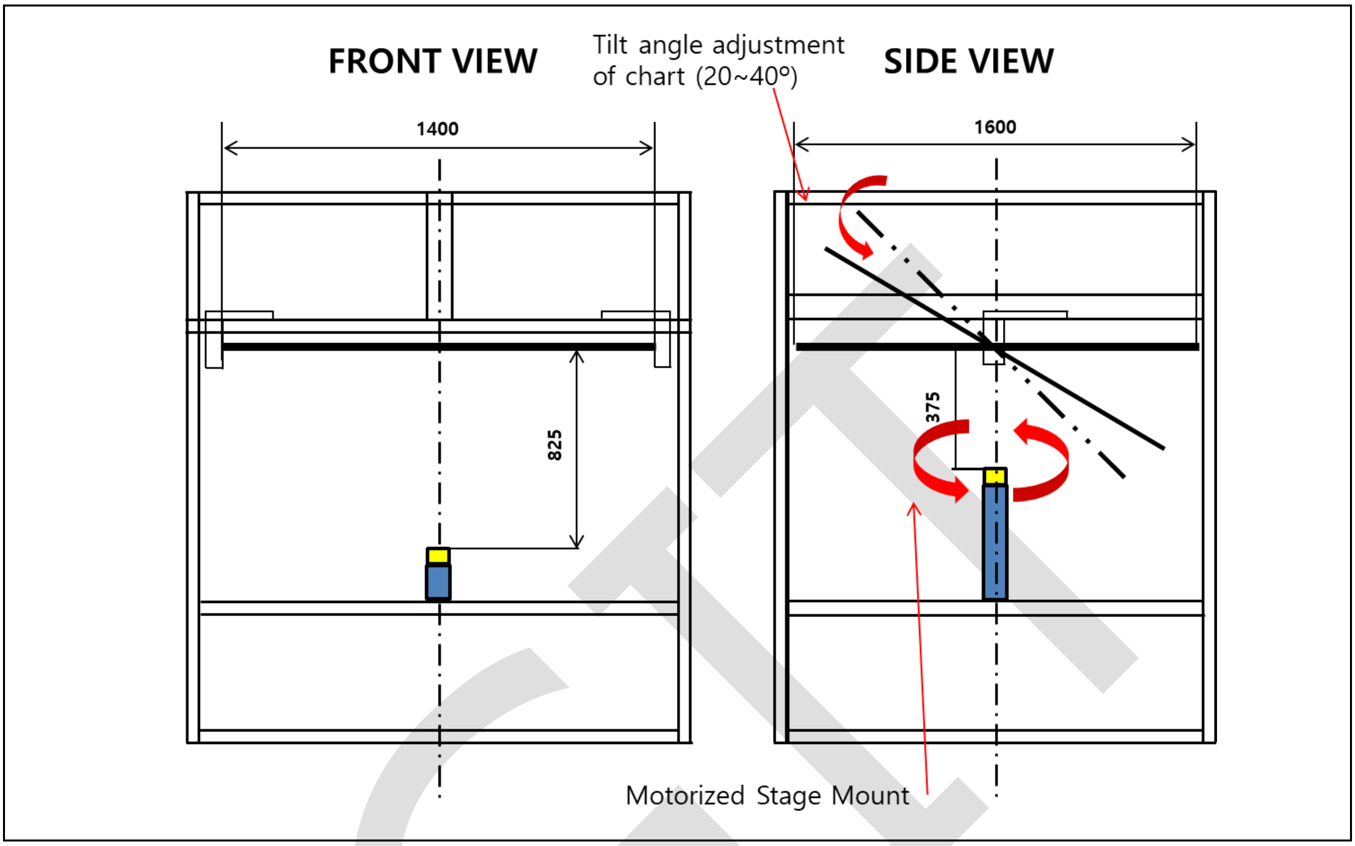

Figure 2.6 Wiggling and Lens-FPPN Calibration Chart. Height adjustable stage mount for module (375 – 825 mm) and an angle adjustable chart (20º – 40º). Height and angle will be fixed for a manufacturing environment.

Table 2.1 The Specification of Wiggling, Lens-FPPN Calibration Chart.

Chart type	Checkerboard
Chart size	1610 mm × 1410 mm
Material (lambertian reflectivity)	White > 80%, Dark > 10%
Square size	40 mm × 40 mm
Square count	40 x 35
Tilt angle adjustment	30º
Angle of Motorized Stage	0º or 180º for wiggling (default is 0º)
Distance from module	375 – 825 mm

When calibrating wiggling, "Wiggling Calibration Board" with the external time-delay as shown in Figure 2.7 has to be used.

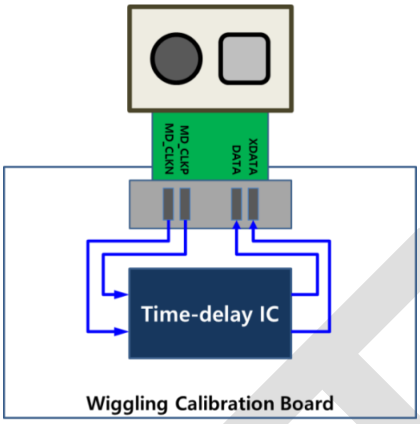

Figure 2.7 The Wiggling Calibration Board.

<u>**Precautions when using the Wiggling Calibration Board**</u>

The time-delay IC on the Wiggling Calibration Board has a delay-variation depending on the temperature. Time delay drift may occur over time, when the internal temperature of the ICs is stabilized. Therefore, sufficient warm-up time may be required when using the Wiggling Calibration Board. We recommend having a warm-up time of at least 15 minutes for stable results. This warm-up time varies with different board design, and is required only at the beginning of the equipment initialization; after the warm-up time, no additional warm-up is required for wiggling calibration process of each individual module.

2.3 **Lens-FPPN**

A geometric calibration uses the standard technique from Zhang [3]. The parameters are usually estimated through homography between 3D and 2D points. By theoretically using at least three different views of a planar checkerboard, the lateral intrinsic camera parameters (principal point, $\mathbf{cc}$, focal length, $\boldsymbol{f}_{\mathbf{C}}$ and lens distortions, $\mathbf{kc}$) are estimated. Lens-FPPN calibration uses at least four different views of a plane due to FPPN.

$\mathbf{K}$ is known as the camera matrix, and defined as follows:

$$\mathrm{K} = \begin{bmatrix} \boldsymbol{f}_{\mathbf{C}}(1) & 0 & \mathbf{cc}(1) \\ 0 & \boldsymbol{f}_{\mathbf{C}}(2) & \mathbf{cc}(2) \\ 0 & 0 & 1 \end{bmatrix}$$

Lens distortion uses the Brown's model as follows:

$$\mathbf{x_d} = \begin{bmatrix} \mathbf{x_d}(1) \\ \mathbf{x_d}(2) \end{bmatrix} = (1 + \mathbf{kc}(1)r^2 + \mathbf{kc}(2)r^4 + \mathbf{kc}(5)r^6)\mathbf{x_n} + \mathbf{dx}$$

where $\mathbf{dx}$ is the tangential distortion vector:

$$\mathbf{x_d} = \begin{bmatrix} 2\mathbf{kc}(1)xy + \mathbf{kc}(4)(r^2 + 2x^2) \\ \mathbf{kc}(3)(r^2 + 2y^2) + 2\mathbf{kc}(4)xy \end{bmatrix}$$

Therefore, the 5-vector $\mathbf{kc}$ contains both radial and tangential distortion coefficients.

Fixed pattern phase noise (FPPN) accounts for pixel-location-dependent phase deviations. The conventional approach is to estimate the FPPN with the aid of a plane and approximate the error with a polynomial fitting. Lens and FPPN calibration can be performed with the same equipment and environment simultaneously because extrinsic parameters of different views of a plane can be obtained in the lens calibration and ground truth (GT) of a plane can be calculated using extrinsic and intrinsic parameters.

2.3.1 **Procedure**

After completing wiggling calibration, lens calibration is performed. Therefore, intrinsic and extrinsic parameters can be obtained. Using extrinsic parameters, FPPN calibration can generate spatial depth offset LUT.

Lens-FPPN Calibration Procedure (as shown in Figure 2.8)

1) Switch from “Time-Delay” to “Direct Connect” using Jumper or Relay Switch

2) Set angles of checker chart as 0°, 90°, 180° and 270°. Additional 10°, 100°, 190° and 280° can be set. If 4 different views of a plane chart are used the number of data have to be increased after evaluating a module.

3) Obtain raw data images in each angle for dual modulation frequencies.

4) Lens calibration is performed using Calibration Library.

a) Calculate intrinsic parameters for each module.

b) Calculate extrinsic parameters for each raw data images.

5) FPPN calibration is performed using Calibration Library.

a) Calculate spatial depth offset.

b) Applies polynomial surface fitting and extracts coefficients

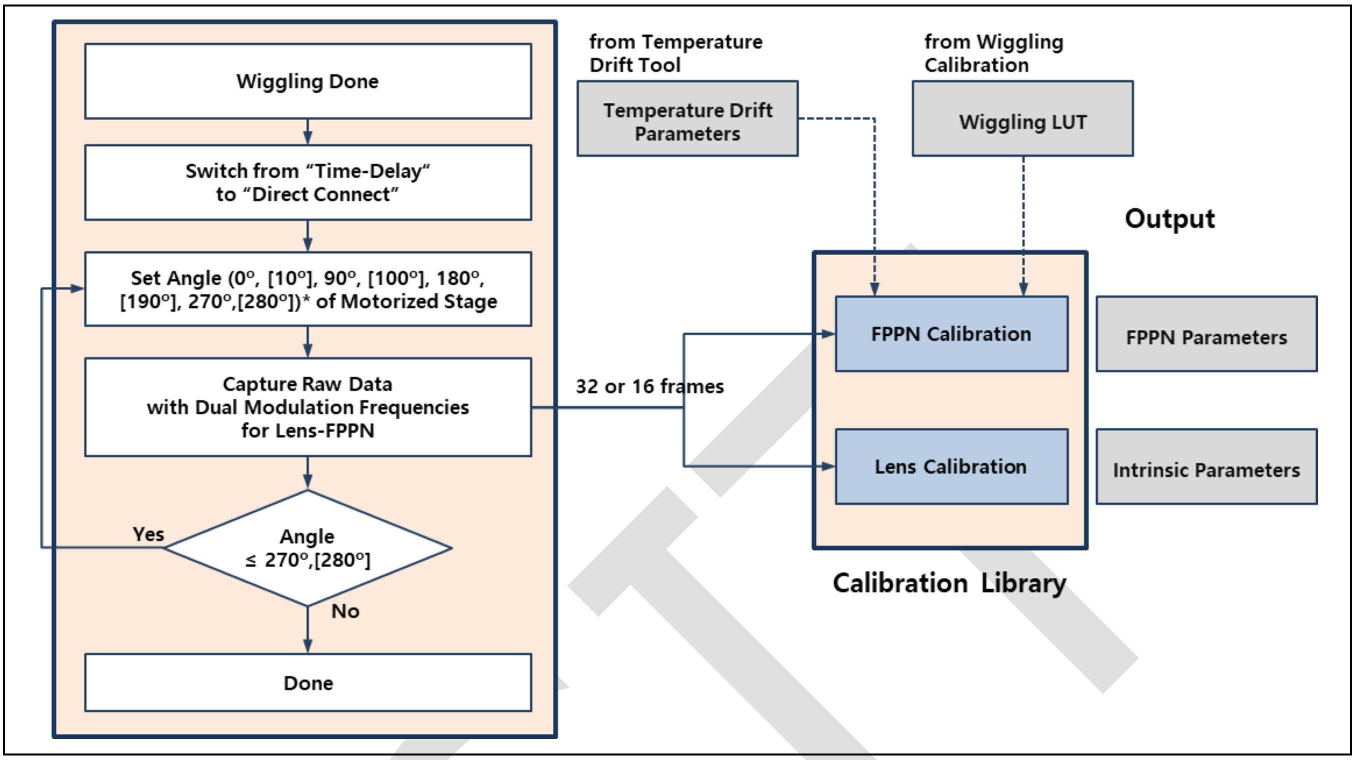

Figure 2.8 Lens-FPPN Calibration Procedure

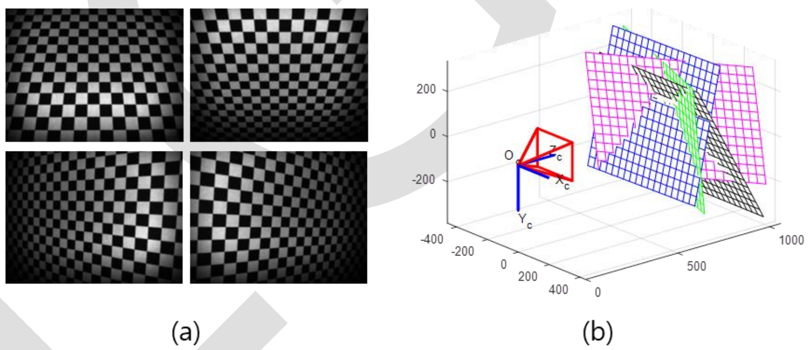

Figure 2.9 Lens Calibration Procedure.

Lens-FPPN Calibration Procedure in detail

FPPN calibration is to compensate pixel-location-dependent error due to the time-delay of the demodulation signal depending on pixel location and the misalignment of VCSEL and sensor (which is a geometric error). Each pixel of a ToF sensor can have different depth offset due to time-delay with respect to the injection of the reference control signal. Also, VCSEL and sensor can be misaligned due to limited space to be located in for small form factor. This misalignment can cause per-pixel depth offset.

As FPPN input data, same checker board images are used with different rotation angles which consists of basically 0°, 90°, 180° and 270°. In order to improve the performance, additional different rotation angles can be added such as 10°, 100°, 190° and 280°. Figure 2.10 show FPPN calibration procedure in detail while using 4 different rotation angle inputs. Ground truth depth image of each input checkerboard image can be generated with extrinsic parameters obtained from lens calibration. Input raw images are converted as measured depth. Low confidence depth data can be removed by using confidence map. Depth offsets are calculated in higher confidence area. Next, remained depth offset images are merged into one offset image.

Since areas of the low-confidence depth pixels are excluded, there are no depth offset data in some spatial positions as shown in Figure 2.11. In order to fill lack of depth offsets, special filtering is applied and then whole depth offsets can be obtained. FPPN depth offset can be estimated by polynomial surface fitting technique. Finally, calibration library calculates the FPPN depth offset LUT.

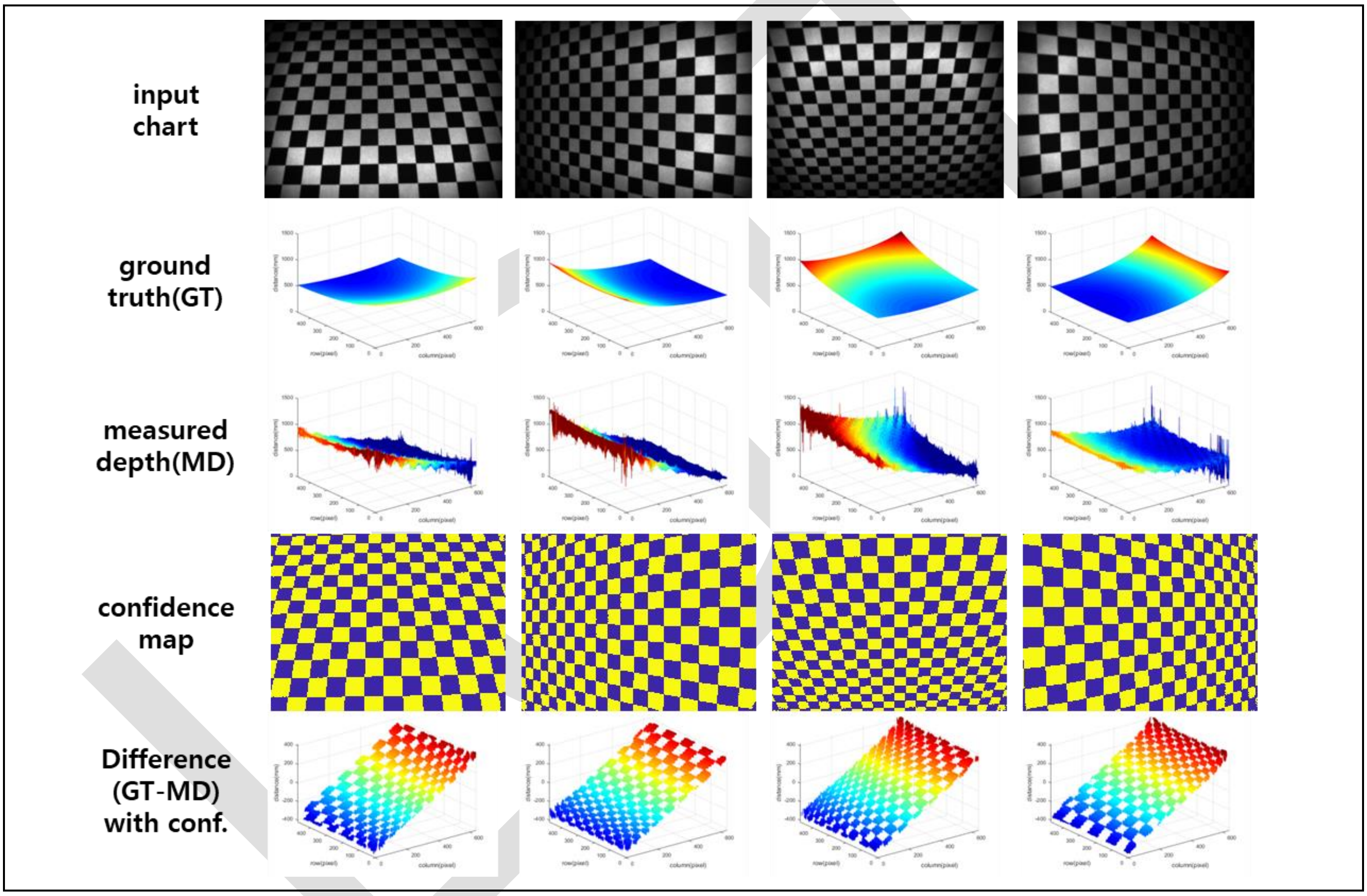

Figure 2.10 FPPN Calibration Procedure; each rotation.

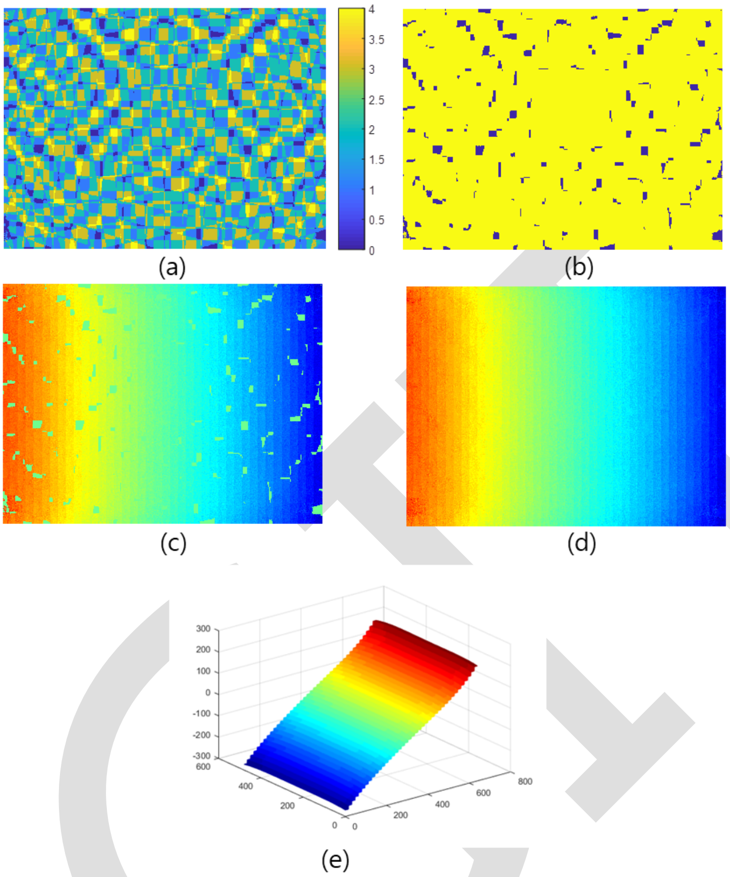

Figure 2.11 FPPN Calibration Procedure; merge all rotations.

2.3.2 **Equipment**

Figure 2.12 and Table 2.2 shows the example of lens-FPPN chart. The specification of lens-FPPN chart can be changed depending on the various conditions. The distance from a module and a plane chart is fixed by monitoring the code to prevent saturation. This distance is about 375 – 825 mm, depending on the optical power of VCSEL, integration time, and chart reflectivity.

When calibrating lens-FPPN, "Lens-FPPN & Temperature Drift Calibration Board" without the external time-delay as shown in Figure 2.3 has to be used.

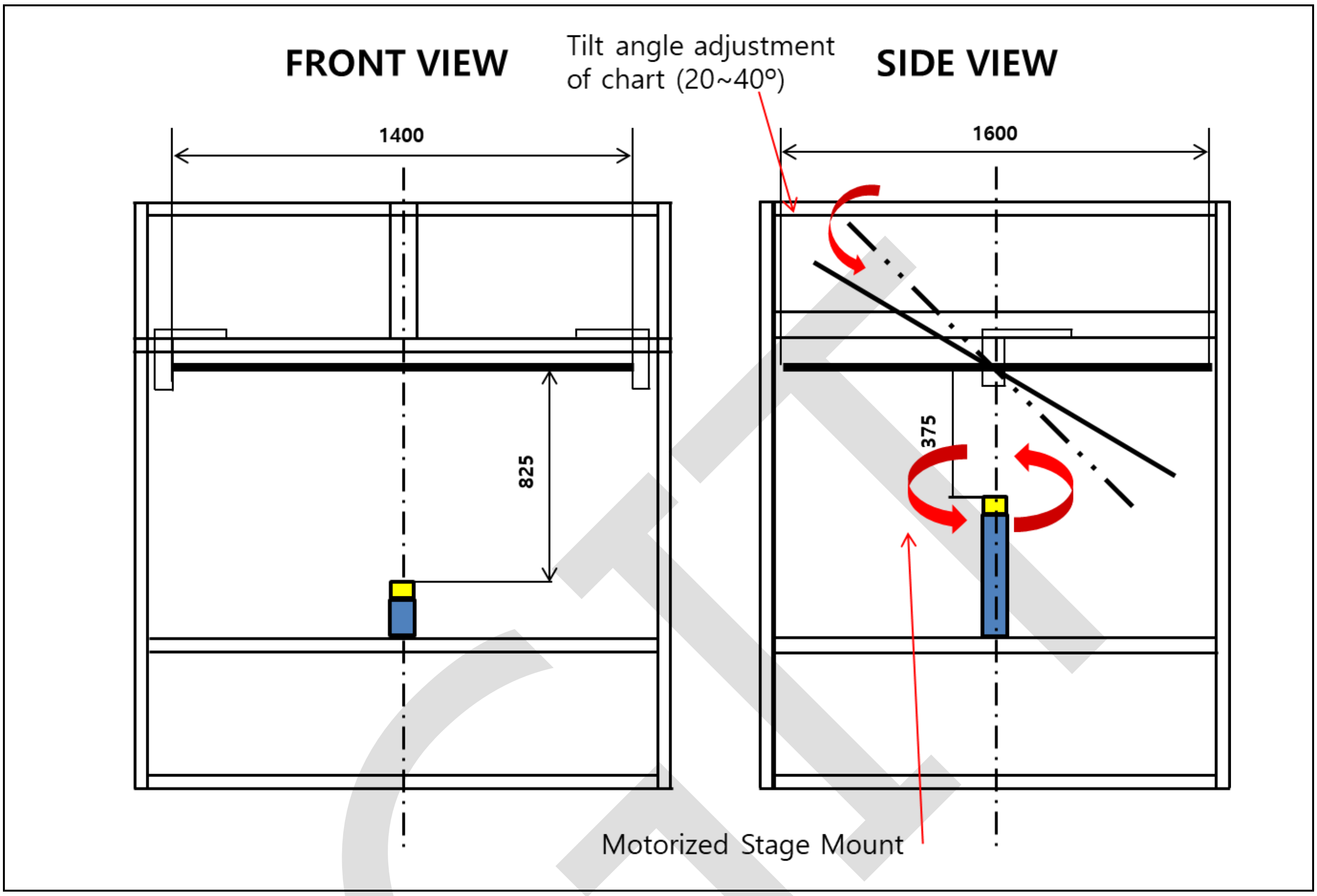

Figure 2.12 Lens-FPPN Calibration Chart. Height adjustable stage mount for module (375 – 825 mm) and an angle adjustable chart (20º – 40º). Height and angle will be fixed for a manufacturing environment.

Table 2.2 The Specification of Lens-FPPN Calibration Chart.

Chart type	Checkerboard
Chart size	1610 mm × 1410 mm
Material (lambertian reflectivity)	White > 80%, Dark > 10%
Square size	40 mm × 40 mm
Square count	40 x 35
Tilt angle adjustment	30º
Angle of Motorized Stage	0º, 90º, 180º, 270º + 10º, 100º, 190º, 280º (added angles depending on performance)
Distance from module	375 – 825 mm

3 Software

3.1 Overview

Software section is to explain usage of software calibration tool and library provided by Samsung Electronics. ToF calibration tools consist of temperature drift tool, calibration library and validation tool, as shown in Figure 3.1. In this section only temperature drift monitoring GUI (graphic user interface) tool for one-time calibration and calibration library for module to module calibration are covered except validation tool.

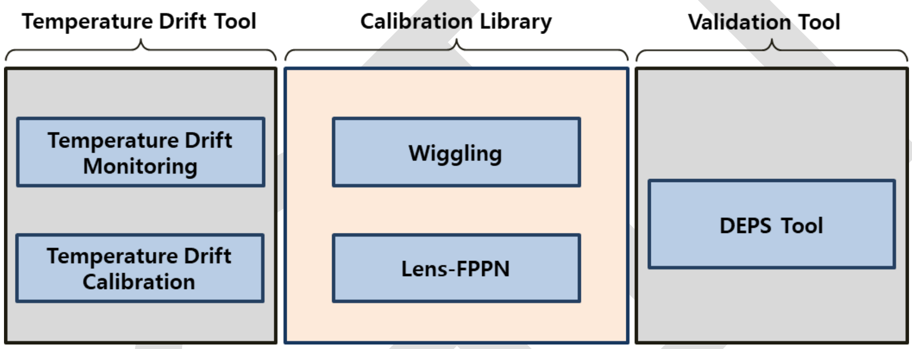

Figure 3.1 Released Softwares; 1) Temperature Drift Tool, 2) Calibration Library, 3) Validation Tool.

This document describes the following softwares.

Version Information

- Temperature Drift Tool: version 1.1.x
- Calibration Library: version 2.1.x

3.2 Temperature Drift Tool

As mentioned in Section 2.1, temperature drift leads to a change in measured distance. In the calibration procedure, the systematic errors could be calibrated with the temperature drift parameters. For these parameters, Temperature Drift Tool calculates the calibration data at measurement of fixed distance.

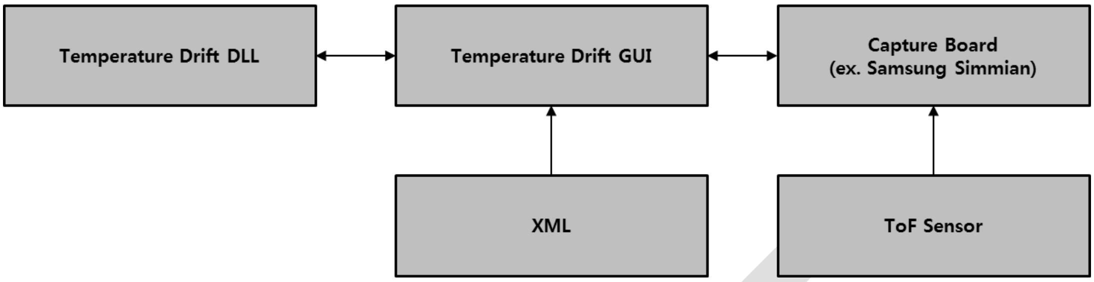

Figure 3.2 Temperature Drift Calibration Block Diagram

The Temperature Drift Tool is composed of data-processing, graphic-user-interface (GUI) and data-capture blocks. The data-processing block is a dynamic link library (DLL) type. Therefore, users can design their own user interface. The usage guide of Temperature Drift DLL will be explained in Section 3.2.1. For example, the Temperature Drift GUI is shown in Fig. 3.3 and controlled by configuration parameters in xml file. Drift Monitor GUI uses Simmain MV3 board and NxSimmian Software for a frame grabber. Your own frame grabber can be combined with Temperature Drift DLL. The Simmian software (NxSimmian) and device drivers must be installed for operation of example, Temperature Drift Monitor GUI.

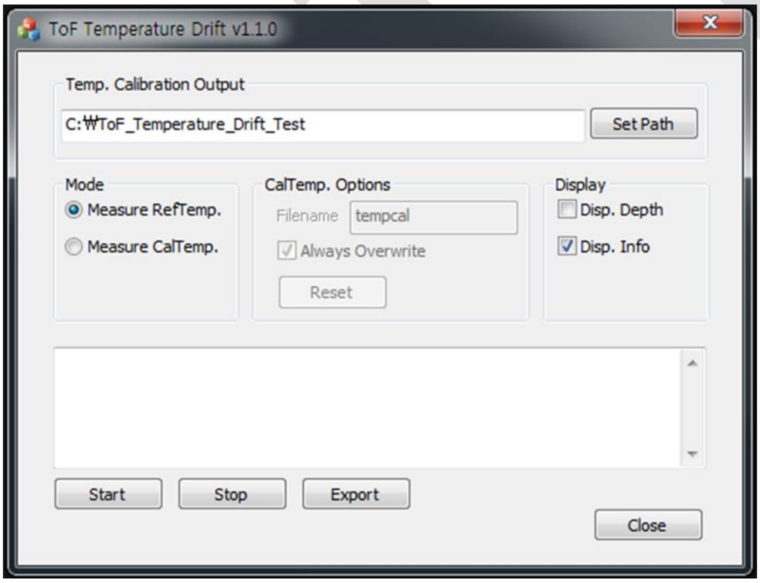

Figure 3.3 Temperature Drift Monitor GUI

3.2.1 **Usage**

The Temperature Drift Monitor options are shown in Fig. 3.4.

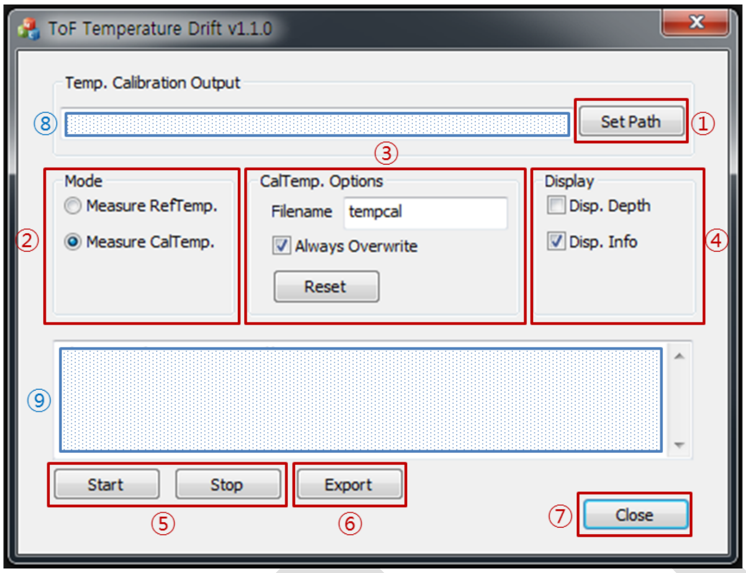

Figure 3.4 Temperature Drift Monitor Options

1. Set Path: Selecting a folder for exporting temperature calibration data. The selected path is shown in the path window (EditControl Num.8). The path of folder should be configured in English only. After all measurements have been completed, the temperature compensation result is saved in this path.

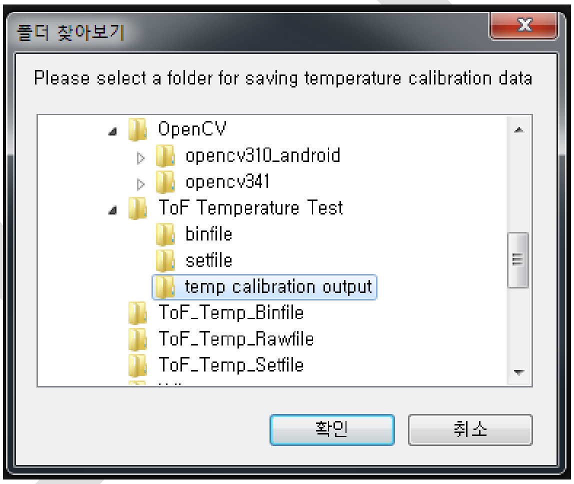

Figure 3.5 Set Path Dialog

2. Mode: Selecting temperature drift monitoring mode. 'Measure RefTemp.' is the mode to measure a reference temperature drift statistics after some warm-up time, t_w. During the warm-up time, distance measurement may not be correct (see Figure 3.6). 'Measure CalTemp.' is the mode to measure a calibrated temperature drift statistics based on reference temperature drift statistics for varying temperature.

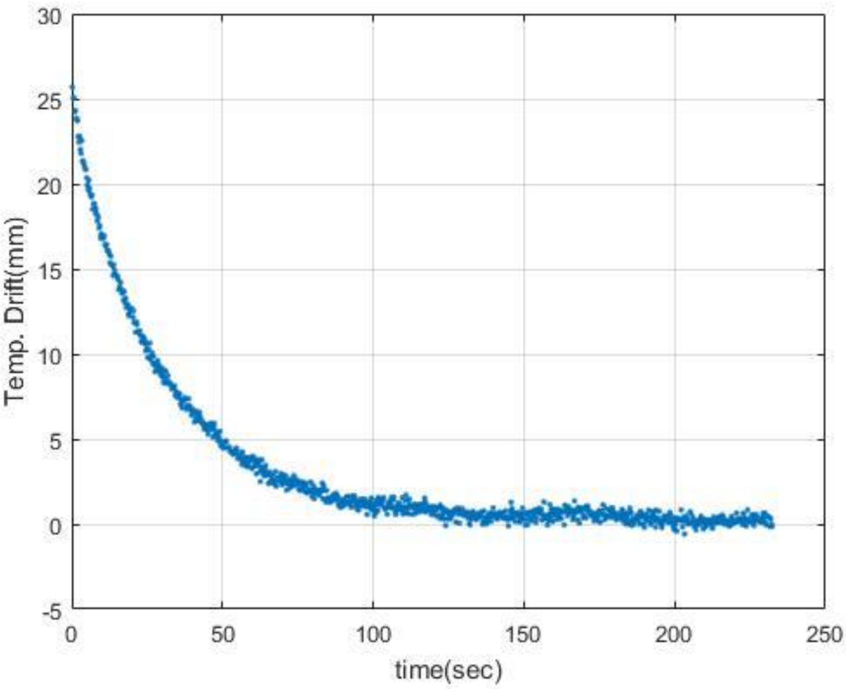

Figure 3.6 Temperature drift offset from reference depth caused by self-induced heating

3. CalTemp. Options: When the 'Measure CalTemp' mode is checked, the options of CalTemp will be enable. The save filename of CalTemp measurement can be set by filename editbox. It can be to check whether the same filename exists by checkbox 'Always Overwrite', and then all previous measurement data could be deleted by the reset button.

4. Display: The checkbox 'Disp. Depth' controls a display window of depth map as shown in Figure 3.7. This checkbox is disable during mode operation. 'Disp. Info' turn on and off about information window (Edit Control Num.9), as shown in Figure 3.8.

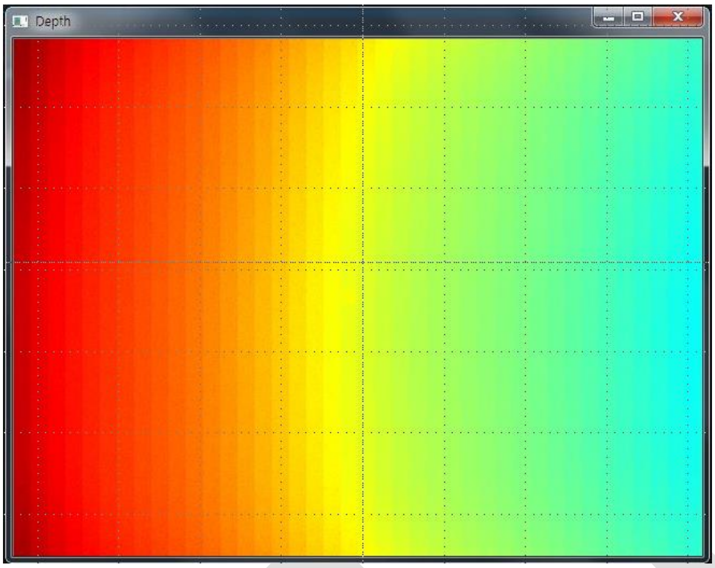

Figure 3.7 Display Depth Map

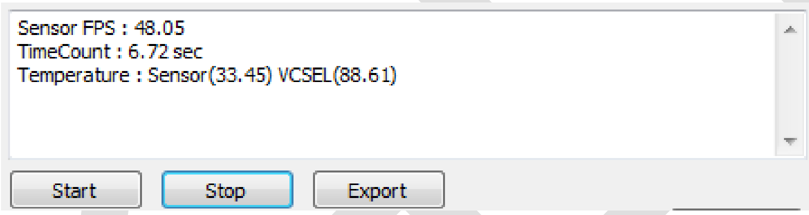

Figure 3.8 Display Information

5. Start and Stop: Start and stop the selected mode operation. After starting a mode, the start button is disabled. The disabled start button is enabled after finishing the mode operation or clicking the stop button.

6. Export: It export the temperature calibration output file with all CalTemp measurement data. The output files will be saved in the path of EditControl Num.8.

7. Close: Close Temperature Drift Monitor tool

3.2.2 **Deliverables**

Below items are deliverables to measure temperature calibration data.

- ToFTemperatureDriftMonitor.exe : execution program
- ToFTemperatureMonitor_DLL.dll : temperature drift monitor library
- ToFTempMonitor.xml : configuration file in XML
- RegisterMap, Setfile

- opencv_world341.dll : OpenCV library

3.2.3 **Temperature Drift Monitor Configuration file**

Temperature Drift Monitor is controlled by configuration parameters in XML files (see Figure 3.9). The path of folder should be configured in English only and all parameters should be enclosed by quotation marks.

```
<TempMonitorXML>
  <!-- Path Config -->
  <RawFilePath>"c:\ToF_Rawfiles\"</RawFilePath>
  <SetFilePath>"c:\ToF_Setfiles\"</SetFilePath>
  <BinFilePath>"c:\ToF_Binfiles\"</BinFilePath>
  <!-- FileName Config -->
  <RegisterMapFileName>"RegsMap.bin"</RegisterMapFileName>
  <SetFileName>"ToFSensor.tset"</SetFileName></TempMonitorXML>
```

Figure 3.9 Temperature Drift Monitor Configuration XML example

- RawFilePath: folder where captured data will be saved
- SetFilePath: folder of setfiles
- BinFilePath: folder where temperature drift monitor data will be saved
- RegisterMapFileName: register map filename
- SetFileName: setfile filename

3.2.4 **Temperature Drift Monitor DLL API**

Temperature Drift Monitor DLL API is class structure supported by Samsung Electronics for processing temperature data. This class includes 10 functions and 7 variables.

```
class TOFTEMPERATUREMONITOR_DLL_API ToF_TemperatureMonitor

{

public:

ToF_TemperatureMonitor();

~ToF_TemperatureMonitor();

// Data
float m_elapsedTime;
float m_sensorTemp;
float m_vcselTemp;
float m_centerDepth[2];
```

```
float m_refFinalCalData[2];
float m_refFinalSensorTemp[2];
float m_refFinalVcselTemp[2];

// Function
void SetRawSize(int row, int col);
void ResetTime();
float ElapsedTime();

bool SetRefData(CString bin_path, CString ref_filename);
bool LoadRefData(CString bin_path, CString ref_filename, CString cal_filename);
void SetRefCount(int refCount);


bool ReadRaw(CString raw_path, int calmode, bool refmode, CString raw_filename);

void GetTempCalVal(CString bin_path, CString save_path, CString* filenames, int
filename_number);

// Debug
bool m_bDispDepth;
void CtrlDispDepth(bool dispDepth);

}
```

Variables

- `m_elapsedTime` : elapsed time after time initialization
- `m_sensorTemp` : sensor temperature value
- `m_vcselTemp` : VCSEL temperature value
- `m_centerDepth` : distance value (0: Freq1 and 1: Freq2)
- `m_refFinalCalData` : distance mean value after warm-up time (0: Freq1 and 1: Freq2)
- `m_refFinalSensorTemp` : sensor temperature mean value after warm-up time (0: Freq1 and 1: Freq2)
- `m_refFinalVcselTemp` : VCSEL temperature mean value after warm-up time (0: Freq1 and 1: Freq2)
- `m_bDispDepth` : control variable for display depth map window

Functions

- void SetRawSize(int row, int col)

  set raw image size
  row – height of raw image
  col – width of raw image

- void ResetTime()

  time initialization

- `float ElapsedTime()`

  return elapsed time after time initialization

- `bool SetRefData(CString bin_path, CString ref_filename)`

  set configurations of reference file to measure the reference data

  bin_path – binary file path
  ref_filename – referece data filename

- `bool LoadRefData(CString bin_path, CString ref_filename, CString cal_filename)`

  set configurations of reference data to load and calibration data to measure

  bin_path – binary file path
  ref_filename – reference data filename
  cal_filename – calibration data filename

- `void SetRefCount(int refCount)`

  set frame numbers for averaging reference temperature data

  refCount – frame numbers to average for reference data

- `bool ReadRaw(CString raw_path, int calmode, bool refmode, CString raw_filename)`

  read raw data file based on raw data file path, mode parameter

  raw_path – raw file path to read (captured raw and embedded file)
  calmode – 0 is 'Measure RefTemp.' to measure the reference data
  and 1 is 'Measure CalTemp.' To measure the calibration data
  refmode – true after warm-up time and false before warm-up time
  raw_filename – captured raw filename to read

- `void GetTempCalVal(CString bin_path, CString save_path, CString* filenames, int filename_number)`

  export the final temperature calibration data to save_path

  bin_path – binary file path
  save_path – temperature calibration output file path
  filenames – measured calibration data filenames
  filename_number – the number of filenames

- `void CtrlDispDepth(bool dispDepth)`

  control display depth map window

`dispDepth` – true for displaying the depth map

This is a block diagram for software development guide with Temperature Drift Monitor API.

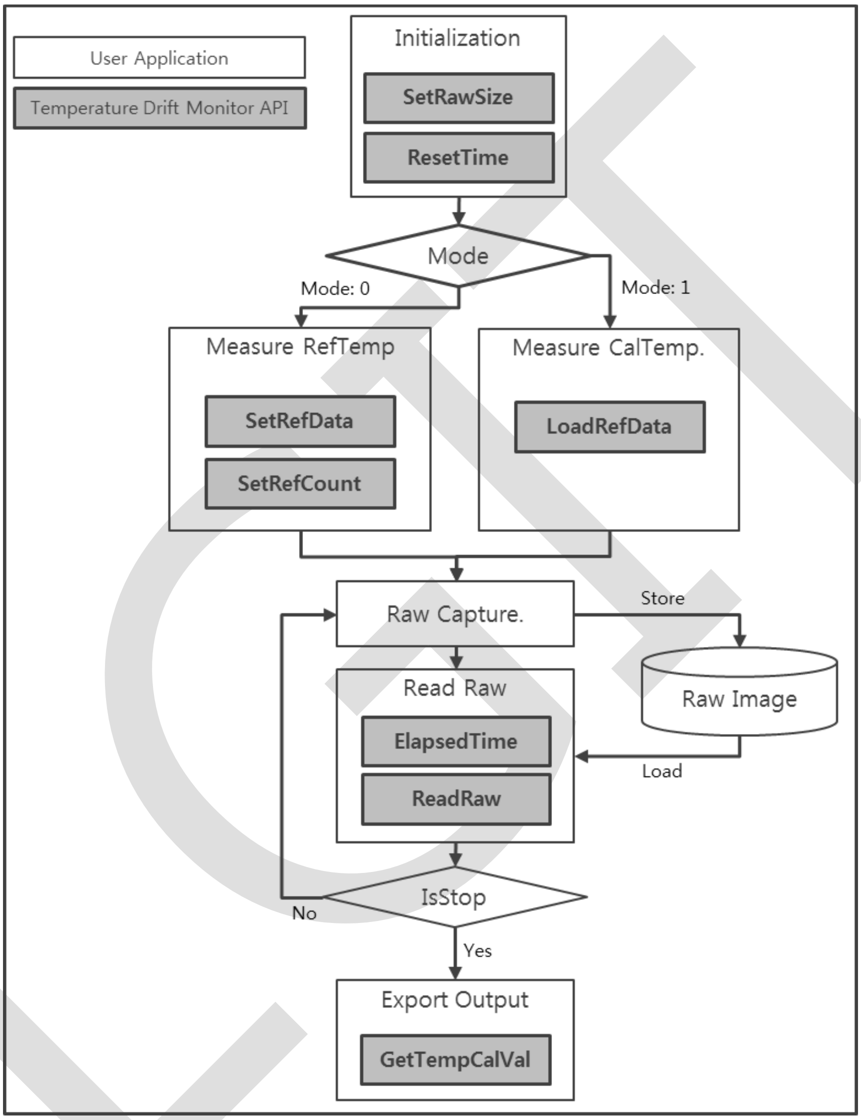

Figure 3.10 Temperature Drift Monitor API Guide

This is example code for Temperature Drift Monitor API.

```
#include <afxstr.h>
#include "ToFTemperatureMonitor_API.h"

int main()
{
```

```
ToF_TemperatureMonitor TempMonitor;
int TemperatureMonitorMode = 0; // mode (0: Measure RefTemp. 1: Measure CalTemp.)
bool bRunnig = true;

int row = 480; // depth height from raw image
int col = 640; // depth width from raw image
int nRefCount = 10; // averaging 10 frames
float fWarmUpTime = 300; // warm-up time (sec)

CString strRawFilePath = _T("c:\\tof_rawfiles\\");
CString strBinFilePath = _T("c:\\tof_binfiles\\");
CString strCalOutputPath = _T("c:\\tof_output\\");

CString strRefFilename = _T("refTempFinalData.bin");
CString strCalFilename = _T("calTempData.bin");
CString strRawFilename = _T("ToF_0000");


if (TemperatureMonitorMode == 0)
{
    /* measure reference temperature drift statistics */
    if (!TempMonitor.SetRefData(strBinFilePath, strRefFilename))
      return -1;

    TempMonitor.SetRefCount(nRefCount);
}
Else if (TemperatureMonitorMode == 1)
{
    /* measure calibrated temperature drift statistics */
    CString strFreq1RefFilename, strFreq2RefFilename;

    int idx = strRefFilename.ReverseFind(_T('.'));
    strFreq1RefFilename = strRefFilename.Left(idx) + _T("_f1.bin");
    strFreq2RefFilename = strRefFilename.Left(idx) + _T("_f2.bin");


     /* check reference data existence */
     if ((GetFileAttributes(strBinFilePath + strFreq1RefFilename) == 0xFFFFFFFF)

     && (GetFileAttributes(strBinFilePath + strFreq1RefFilename) == 0xFFFFFFFF))
     {

        /* error */
       printf("Could not find the reference data files");
       return -1;

     }

     else
       TempMonitor.LoadRefData(strBinFilePath, strRefFilename, strCalFilename);

  /* raw file size */
  TempMonitor.SetRawSize(row, col);

  /* init. time */
```

```
    TempMonitor.ResetTime();

    bool bRef = false; // warm-up time flag to measure reference data

    while (bRunnig)
    {

        if (!bRunnig) break;

        /* get current time */
        float elapsed_time = TempMonitor.ElapsedTime();

        /* user capture code */
        // TODO : place code here to capture raw file
        // user client capture code with raw filename(strRawFilename)

        /* check warm-up time */
        if (elapsed_time > fWarmUpTime && TemperatureMonitorMode == 0)
            bRef = true;

         /* read raw file */
        TempMonitor.ReadRaw(strRawFilePath,TemperatureMonitorMode,bRef,strRawFilename);

        /* user read variables */
        // TODO : place code here to read variables
        // Sensor Temperature : TempMonitor.m_sensorTemp
        // Vcsel  Temperature : TempMonitor.m_vcselTemp
        // Depth : TempMonitor.m_centerDepth[0], TempMonitor.m_centerDepth[1]

         if (!bRunnig) break;

     }


     /* make measured calibration data list */
     // The measured calibration data is stored in subfolder ('caldata') of
     // binary file path. We can export the final temperature calibration output
     // by filename list in the subfolder.

     vector<CString> vecCalFile;
     CString findpath = strBinFilePath + _T("caldata\\");
     CString findfile = _T("*.bin");

     long handle;
     int result = 1;
     _finddata_t fd;

     handle = _findfirst((CStringA)(findpath + findfile), &fd);

     while (result != -1 && handle != -1)
     {
         vecCalFile.push_back(CString(fd.name));
         result = _findnext(handle, &fd);
     }
```

```
    _findclose(handle);

    /* export calibration output */
    TempMonitor. GetTempCalVal(strBinFilePath, strCalOutputPath, &vecCalFile[0],
vecCalFile.size());

    return 0;
}
```

3.2.5 Temperature Drift Monitor Output

Temperature Drift Monitor tool generates outputs as binary format of temperature drift statistics. The some file name can be changed by user.

- `tempcal.bin`

  temperature drift data (The filename can be changed. It is stored for each frequency)
  (float elapsed_time, float sensor_temperature, float vcsel_temperature, float depth)

- `refTempFinalData.bin`

  temperature drift data after warm-up time (mean value of data the number of `SetRefCount set value`)
  (float sensor_temperature_mean, float vcsel_temperature_mean, float depth_mean)

- `temp.bin`

  temperature calibration output
  (float calibration_val)

3.3 Calibration Library

Calibration library is common dynamic library (*.dll) developed in C++. Since calibration library internally uses OpenCV library's functions and data structures, OpenCV dynamic library (currently 3.4.1 version) is included in deliverables. Also, two sets of calibrarion library will be provided according to operating systems x64 and x86.

3.3.1 Deliverables

Below items are deliverables to calibrate ToF sensor.

(1) Opencv_world341.dll : OpenCV library

(2) ToFCalParam.xml : calibration opearation configuration file in XML

(3) ToF_Calibration_DLL_x64.dll(.lib)/ ToF_Calibration_DLL_x64.lib(.lib) : calibration dynamic libraries for x64

(4) ToF_Calibration_DLL_Win32.dll(.lib) / ToF_Calibration_DLL_Win32.lib(.lib): calibration dynamic libraries for x86

(5) ToF_Calibration_EXE_x64/ToF_Calibration_EXE_Win32 : execution example program

(6) ToF_Calibration_API.h : API definition C++ header file

(7) ToF_Calibration_EXE.cpp : Execution program example inC++ source file

(8) ToF_Calibration_EEPROM_MAP.pdf : Memory map of calibration output result

(9) Temp_bin_default : Temperature drift binary files

3.3.2 **API**

Only one main API is needed to execute calibration for module to module(M2M) calibration by calling in execution program.

```
TOF_CALBRATION_API int Calibrate_ToF()
```

Execute ToF calibrations. Returns success (0) or fail(-1)

Parameters :

- `TOF_CALBRATION_API :` defines as `__declspec(dllimport)` for dynamic libraray usage

Calibration execution example is given below. Firstly function `Calibrate_ToF()` calling is carried out. And then calibration execution result can be monitored by checking return value.

```
#include <iostream>
#include "ToF_Calibration_API.h"

using namespace std;

int main()
{
	int ret = 0;
	ret = Calibrate_ToF();
	if(ret == 0)
		cout<<endl<< ">>>>>>> Tof Calibration is done <<<<<<<" <<endl;
	else
		cout<<endl<< ">>>>>>> Tof Calibration is fail <<<<<<<" <<endl;

	return ret;
}
```

3.3.3 **Calibration Configuration file**

In order to control and set the proper calibration environmnent, ToFCalParam.xml is provided. ToFCalParam.xml includes input/output interfaces and specific paramaters in each calibration module.

```xml
<!-- Common Calibration Parameters -->
<COMM_TempCalFilePath>".\Temp_bin_default\"</COMM_TempCalFilePath>
<COMM_RawFilePath>".\DB\"</COMM_RawFilePath>
<COMM_CalFilePath>Calibration\</COMM_CalFilePath>
<COMM_ImageWidth>640</COMM_ImageWidth>
<COMM_ImageHeight>480</COMM_ImageHeight>
<COMM_ModulationFrequency1>100</COMM_ModulationFrequency1>
<COMM_ModulationFrequency2>80</COMM_ModulationFrequency2>
<!-- Wiggling Calibration Parameters -->
<WIGG_RawFilePath>wiggling\</WIGG_RawFilePath>
<WIGG_RawFilePrefix>wiggling_1ms_ag1_</WIGG_RawFilePrefix>
<WIGG_DisStr>0</WIGG_DisStr>
<WIGG_DisEnd>1392</WIGG_DisEnd>
<WIGG_DisStep>72</WIGG_DisStep>
<WIGG_DisCap>550</WIGG_DisCap>
<!-- Lens & Fppn Calibration Parameters -->
<LENS_RawFilePath>lens_fppn\</LENS_RawFilePath>
<LENS_RawFilePrefix>cal_1ms_ag1_</LENS_RawFilePrefix>
<LENS_Num>8</LENS_Num></CalParameter>
```

Figure 3.11 Calibration Configuration XML example

3.3.3.1

3.3.3.2 Common Calibration Parameters

Common paramters for all calibration modes can be set.

COMM_TempCalFilePath: Temperature drift tool binary file location in English. Note that temperature drift binary files must be included in this folder.

- Default: "..\Temp_bin_default\"

COMM_RawFilePath: common calibration input file location in English

- Default: "..\ DB\"

COMM_CalFilePath: calibration result output file location. Internally this location is added to COMM_RawFilePath in English

- Default: "Calibration\" (actual path is set as "..\DB\Calibration\")

COMM_ImageWidth: final depth output image width

- Default : 640

COMM_ImageHeight: final depth output image height

- Default : 480

COMM_ModulationFrequency1: first modulation frequency value in MHz

- Default : 100

COMM_ModulationFrequency2: second modulation frequency value in MHz

- Default : 80

3.3.3.3 Wiggling Calibration Paramters

Wiggling calibration parameters can be set.

WIGG_RawFilePath: wiggling calibration input file location. Internally this location is added to COMM_RawFilePath.

- Default: “wiggling\” (actual path is set as “..\ DB\wiggling\”)

WIGG_RawFilePrefix: starting input raw file name

- Default : “wiggling_1ms_ag1_” (actual raw file name is set as “wiggling_1ms_ag1_~~.raw”)

WIGG_DisStr: starting control code setting of the time-delay circuit

- Default : 0

WIGG_DisEnd: ending control code setting of the time-delay circuit

- Default : 1280

WIGG_DisStep: increment of time-delay control code

- Default : 64

WIGG_DistCap: initial estimate of the distance between the sensor and the captured chart in mm

- Default : 550
-

3.3.3.4 Lens & FPPN Calibration Paramters

Lens calibration parameters can be set.

LENS_RawFilePath: lens calibration input file location. Internally this location is added to COMM_RawFilePath.

- Default: “lens_fppn\” (actual path is set as “..\ DB\ lens_fppn\”)

LENS_RawFilePrefix: starting input raw file name

- Default : “cal_1ms_ag1_” (actual raw file name is set as “cal_1ms_ag1_~~.raw”)

LENS_Num: total calibration input image number same as converted depth image number

- Default : 8 (actual raw file number is same as LENS_Num * 4 in 4 Tab, shuffle and dual frequency mode)

3.3.4 **Calibration Result Output**

Calibration library generates each calibration block result and merge full calibration results in a binary format. Each calibration block results can be used internally. Also full calibration results can be used to write in a memory such as EEPROM on the camera module.

Note that in order to build full calibration binary, temperature calibration binaries from the temperature drift tool should be included in same calibration folder previsouly set in XML.

3.3.4.1 Calibration output in each calibration block

temp_f1.bin/ temp_f2.bin: temperature calibration results in binary format for each modulation frequency

wigg_f1_tbl.bin/ wigg_f2_tbl.bin: wiggling calibration results in binary format for each modulation frequency

fppn_f1_tbl.bin/ fppn_f2_tbl.bin: FPPN calibration results in binary format for each modulation frequency

- lens.bin: lens calibration result in binary format

3.3.4.2 Calibration full output

_ToF_cal.bin : merged full calibration result according to memory map.

3.4 **DEPS Tool**

DEPS tool and document will be updated.

References

[1] https://en.wikipedia.org/wiki/Thermoelectric_cooling

[2] https://www.onsemi.com/pub/Collateral/NB6L295-D.PDF

[3] Z. Zhang, "Flexible Camera Calibration by Viewing a Plane from Unknown Orientations," ICCV99, vol. 1, pp. 666-673, 1999.5.

[4] D. Brown, "Decentering distortion of lenses". Photogrammetric Engineering. 32 (3), pp.444-462, 1966.

[5] https://opencv.org/