Instruction manual for the calibration software

by ***pmd****technologies*

Abstract

This document describes a step by step instruction manual on how to tune the calibration software parameters for new camera devices, new lenses, new illumination sources or new calibration boxes.

Table of Contents

Abstract 1
Table of Contents 1
1. Data acquisition 2
2. Adjustment of the calibration settings: Box parameters 2
2.1. General parameters 2
2.2. Box setup parameters 3
2.2.1. LED Box 3
2.2.2. Fiber box 4
3. Initial calibration only with LED box data 5
3.1. Amplitudes 5
3.2. Defect pixels 5
3.3. Inspection of all limits after successful calibration 5
4. Add wiggling calibration by using LTS data 9
5. Add individual temperature drift data 9
6. Calibrate fiber box with golden sample data 10
7. Final limit definition 10
Document History 11

1. Data acquisition

Acquire raw data from the LED and fiber box as well as with the linear translation stage (LTS). This data has to conform to the rules of document “Cal-4-1-AN-Calibration_Boxes_Data_Acquisition_and_Processing.pdf”. The first try for the exposure time setting can be

- either extrapolated from former devices and box setups
- or by using the exposure time from the use case that should later work in the device.

2. Adjustment of the calibration settings: Box parameters

A configuration file contains various kinds of parameters:

- general parameters
- box setup parameters
- calibration configuration parameters
- limit parameters

2.1. General parameters

First of all, all general parameters, that are independent of the ToF system to be calibrated, have to be set.

- **debug**
  For configuration **debug = 1** has to be set. This creates plots during calibration to find any anomalies in the data.

- **OutBaseDir**
  Set the output directory where the calibration file is stored

- **LogOutDir**
  Set the output directory where the log file (contains all calibration limits, data values and a pass/fail number) is stored

- **Calibration_Data_Format**
  Define the output format:
  Calibration_Data_Format = 1: plain SPC data: (no header)
  Calibration_Data_Format = 2: EEPROM binary V1: (HeaderV3,SPC,HeaderV5,Public)
  Calibration_Data_Format = 3: EEPROM binary V2: (HeaderV6+DataBlocks)
  Calibration_Data_Format = 4: EEPROM binary: HeaderV3+SPC
  only for Calibration_Data_Format = 3: file name for "ProductCode" data block
  ProductCodeFile = '<Config_Dir>\data\ProductCode.bin':

- **Calibration_Data_Size_Limit**
  Define the maximum size (in bytes) for the calibration data (e.g. limited by EEPROM size). For 1 Mbit EEPROM size **Calibration_Data_Size_Limit = 12500** is set.

2.2. Box setup parameters

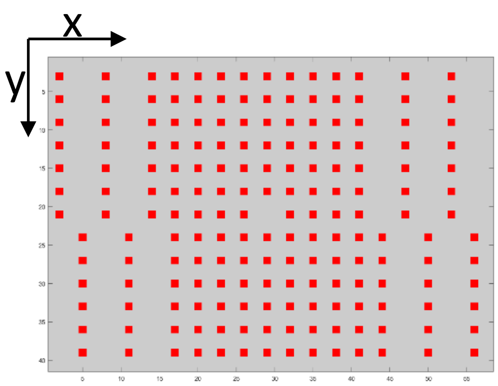
These parameters are independent of the camera settings and do not need to be changed when the camera settings are changed.

2.2.1. LED Box

- **Reflectivity**
  The reflectivity of the white paper foil of the LED box. Current setting is Reflectivity = 0.90 (adaption might by necessary if a different foil is used).

- **LC.pattern_mask**
  The infrared LEDs are positioned in a defined grid. **1** define an LED on this specific position, **0** defines a 'free' position. The red center LED is marked as 0, because it is invisible for the ToF sensor. An example grid is shown below:

```
LC.pattern_mask = [...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0; ...
   1,0,1,0,1,1,1,1,1,0,1,1,1,1,0,1,0,1,0; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1; ...
   0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1];
```



- **LC.pattern_spacing**
  The distance between two drill holes in the grid is currently 3cm (**LC.pattern_spacing = 0.03)**

- **LC.pattern_center**
  In the center of the LED pattern the red LED is positioned. It is necessary for the alignment of the LED box but invisible for the ToF sensor itself.
  The first coordinate is the x axis position (0 indicated counting) of the center LED (in the below example the tenth LED in x direction times LC.pattern_spacing), the second coordinate corresponds to the y axis position (0 indicated counting). The third coordinate is the distance between the LED plate and the entrance pupil of the lens of the ToF device (in meters).
  LC.pattern_center = [9*0.03, 6*0.03, 0.1758];

- **LC.SpotSize**
  The size of the LED spots to be detected. Standard value is **LC.SpotSize = 5** and does not need to be adapted.

- **LC.SpotDetector_type**
The detector type can be set to **1** (spot detection) or **2** (threshold detection). Standard detector type is **LC.SpotDetector_type = 2** and does not need to be adapted.

- **LC.SD.noise_threshold**
The threshold value between a spot and background noise. The standard value is **LC.SD.noise_threshold = 100** and only needs to be adapted if new kind of LEDs are used that differ from older ones concerning the properties.

- **LC.lens_model**
Define the model used for lens calibration:
LC.lens_model = 1: polynomial lens model (standard model)
LC.lens_model = 2: fisheye lens model (needs a different number of lens parameters)
The fisheye model is not yet implemented in the data processing pipeline.

- **LC.lens_parameter**
Define initial parameters for the lens that is used. Those parameters work as starting parameters for the fitting algorithm, thus <u>have to be chosen very accurate for each new lens</u> (independent from all other parameters like chip, source of illumination, etc.). For the polynomial lens model nine parameters are necessary:

  focal length in x direction, in y direction, position of the center LED in x direction, in y direction and five different polynomial and rotation parameters k1, k2, p1, p2, k3.

  An example then looks like this:
  LC.lens_parameter = [113.02, 113.02, 111.5, 85.5, -0.2641, 0.0952, 0, 0, -0.0145];

  Make sure to adapt the limits **LC.fx_limits**, **LC.fy_limits**, **LC.cx_limits**, **LC.cy_limits**, **LC.rvec_x_limits** so that they fit to the above chosen lens parameters. fx and fy can be taken from the data sheet, cx and cy should lie in the center of the ROI.

- **LC.spot_count_limits**
The total number of LEDs that have to be detected from the algorithm. Upper limit is given by the maximum number of LEDs that are used in the pattern. The lower limit has to be chosen in that way that enough LEDs are available for a proper lens angle calibration.

2.2.2. Fiber box

The parameters for the fiber box are not needed at this point of the calibration process and thus do not have to be set now. For fiber box calibration and parameter settings refer to section 0.

3. Initial calibration only with LED box data

For the initial calibration only data from the LED box is needed. Thus the software uses global wiggling parameters and global temperature drift parameters for a first calibration run. The log file can be used to check the limits.

In case the calibration was not successful the log file can be used to view and - in case it is necessary - to change limits. The log file looks like this:

0005006; cy; px; 85.273; 77; 95; 1

An internal number of the calibration point is followed by the parameter name (**cy**), a unit (**px**), the measured value (**85.237**), the lower (**78**) and upper limit (**95**), and a pass (**1**) or fail (**0**) criteria.

3.1. Amplitudes

A failing calibration due to low amplitudes or too high noise (in correlation to the limits) lead to an increase of the exposure time, too high amplitudes lead to a decrease of the exposure time.

3.2. Defect pixels

Too many defect pixels (of which reason ever) in the outer range can be caused by a low illumination signal or relative illumination (vignetting). This can be avoided by reduction of the inclination angle (declared in degree):

DP.Inclination_angle = [0, 60] (example)

3.3. Inspection of all limits after successful calibration

After the calibration calculation successfully generated the calibration file, all limits have to be checked again to make sure that all the limits are neither too narrow nor too widen due to the adaption. An **example** limit adaption is shown below, detailed information about the different calibration points is described in “CAL-14-1-AN-Calibration_Limits”.

- **LC.pixel_mapping_limits = [0,10]**
  The maximum distance (in pixels) between real position of LED spots and calculated position. Lower limit is 0, upper limit should be maximum 10.
- **LC.spot_count_limits = [100,181]**
  Limit for the number of LED spots that are detected. Upper limit is given by the total number of LEDs built in. Missing lines of LEDs at the sides of the pattern reduce the total number of LEDs detected. The number of maximum missing LED pattern lines that are tolerated for a proper lens calibration define the lower limit.
- **LC.fx_limits = [108,118]**
  Limit for the focal length in x direction (should be uniformly around starting parameter value →refer to lens data sheet). First try can be +/- 5, statistical data can change the range later.
- **LC.fy_limits = [108,118]**
  Limit for the focal length in y direction (should be uniformly around starting parameter value →refer to lens data sheet). First try can be +/- 5, statistical data can change the range later.
- **LC.cx_limits = [108,118]**
  Limit for the center pixel in x direction (should be uniformly around the center pixel). First try can be +/- 5, statistical data can change the range later.

- **LC.cy_limits = [80,90]**
  Limit for the center pixel in y direction (should be uniformly around the center pixel). First try can be +/- 5, statistical data can change the range later.
- **LC.rvec_x_limits = [-0.05,0.05]**
  Limit for the x value of the rotation vector (should be uniformly around 0 and match for x, y, z value).
- **LC.rvec_y_limits = [-0.05,0.05]**
  Limit for the y value of the rotation vector (should be uniformly around 0 and match for x, y, z value).
- **LC.rvec_z_limits = [-0.05,0.05]**
  Limit for the z value of the rotation vector (should be uniformly around 0 and match for x, y, z value).

- **DP.DarkCurrent = [-10, 10]**
  Limit for the maximum dark current in the ToF chip in DN/s (for the worst performing pixel). Should be around 0 and positive. As the absolute value of the dark current is very low, also small negative values are acceptable. Pixels outside this limit are flagged as defect pixels.
- **DP.FPN = 2047+[-200,200]**
  Limit for FPN. It is symmetric around 2^11-1 = 2047. Upper and lower limit should be symmetric. Pixels outside this limit are flagged as defect pixels.
- **DP.DME1 = [0.5, 1.1]**
  Limit for the dynamic mixing efficiency for the first modulation frequency. Values below 0.5 describe bad pixels (bad ToF chip mixing performance), values much higher than 1.0 refer to bad S/N ratio pixels. Pixels outside this limit are flagged as defect pixels.
- **DP.DME2 = [0.5, 1.1]**
  Limit for the dynamic mixing efficiency for the second modulation frequency. Values below 0.5 describe bad pixels (bad ToF chip mixing performance), values much higher than 1.0 refer to bad S/N ratio pixels. Pixels outside this limit are flagged as defect pixels.
- **DP.FPPN1 = [-10, 10]**
  Limit for the FPPN values (in m) for the first modulation frequency. Currently this limit is not implemented (therefore wide limits of +/- 10m).
- **DP.FPPN2 = [-10, 10]**
  Limit for the FPPN values (in m) for the second modulation frequency. Currently this limit is not implemented (therefore wide limits of +/- 10m).
- **DP.Amplitude1 = [30, 2500]**
  Limits for the amplitude value (in DN) for the first modulation frequency. Lower limit should be around 30 for minimum S/N ratio, upper limit is restricted to saturation. Pixels outside this limit are flagged as defect pixels.
- **DP.Amplitude2 = [30, 2500]**
  Limits for the amplitude value (in DN) for the second modulation frequency. Lower limit should be around 30 for minimum S/N ratio, upper limit is restricted to saturation. Pixels outside this limit are flagged as defect pixels.
- **DP.Intensity = [10, 2500]**
  Limits for the intensity value (in DN). Lower limit should be around 30 for minimum S/N ratio, upper limit is restricted to saturation. Pixels outside this limit are flagged as defect pixels.
- **DP.Inclination_angle = [0, 60]**
  Limit for the inclination angle of the field of view (in degrees). Lower limit is 0 (no center pixels flagged out), upper limit is defined by the OEM's spec.

- **TP.defect_pixel_count = [0, 40]**
  Limit for the number of defect pixels allowed. Lower limit is 0, upper limit defined by the OEM's spec (e.g. as a total number of valid pixels) and should be in the scale of 40.
- **TP.defect_cluster_size = [0, 2]**
  Limit for the cluster size of defect pixels. Lower limit is 0, upper limit should be small, large clusters create big 'dark spots' in the image.
- **TP.valid_pixel_count = [37000, 40000]**
  Limit for the total number of valid pixels. Upper limit is defined by the total number of pixels available on the ToF chip, lower limit is defined by the OEM's spec.
- **TP.DarkCurrent_mean = [-0.1, 0.1]**
  Limit for the mean dark current in the ToF chip in DN/s. Should be around 0 and positive. As the absolute value of the dark current is very low, also small negative values are acceptable.
- **TP.FPN_mean = [2000, 2100]**
  Limit for the mean value of FPN. Limit should be roughly symmetric around 2^11-1 but narrower than DP.FPN
- **TP.PhaseSTD1_mean = [0.0001, 0.02]**
  Limit for the mean phase standard deviation of the first modulation frequency (in m). Lower limit can be arbitrary near 0, but positive. Upper limit is defined by the spec and should not exceed 0.02m.
- **TP.PhaseSTD2_mean = [0.0001, 0.02]**
  Limit for the mean phase standard deviation of the second modulation frequency (in m). Lower limit can be arbitrary near 0, but positive. Upper limit is defined by the spec and should not exceed 0.02m.
- **TP.PhaseSTD1_max = [0, 1.25]**
  Limit for the maximum phase standard deviation of the first modulation frequency (in m). Lower limit is 0. Upper limit is defined by the spec.
- **TP.PhaseSTD2_max = [0, 1.25]**
  Limit for the maximum phase standard deviation of the first modulation frequency (in m). Lower limit is 0. Upper limit is defined by the spec.
- **TP.Amplitude1_mean = [200, 1400]**
  Limit for the mean amplitude for the first modulation frequency. Lower limit is defined by 200 for a good S/N ratio, upper limit is defined by 1400 to avoid pixel saturation.
- **TP.Amplitude2_mean = [200, 1400]**
  Limit for the mean amplitude for the second modulation frequency. Lower limit is defined by 200 for a good S/N ratio, upper limit is defined by 1400 to avoid pixel saturation.
- **TP.Amplitude1_max = [200, 1400]**
  Limit for the maximum amplitude for the first modulation frequency. Lower limit is defined by 200 for a good S/N ratio, upper limit is defined by 1400 to avoid pixel saturation.
- **TP.Amplitude2_max = [200, 1400]**
  Limit for the maximum amplitude for the second modulation frequency. Lower limit is defined by 200 for a good S/N ratio, upper limit is defined by 1400 to avoid pixel saturation.
- **TP.FPPN1_std = [0, 0.1]**
  Limit for the standard deviation of the FPPN for the first modulation frequency. Lower limit is defined by 0 (no standard deviation), upper limit at 0.1m.
- **TP.FPPN2_std = [0, 0.1]**
  Limit for the standard deviation of the FPPN for the first modulation frequency. Lower limit is defined by 0 (no standard deviation), upper limit at 0.1m.

- **TP.AmplitudeWigglingOffset1 = [0.6, 1.0]**
  Limit for the amplitude wiggling offset for the first modulation frequency. Offset limits are set between 0.6 and 1.0 (smooth limits that can be adapted for each batch of ToF modules).
- **TP.AmplitudeWigglingOffset2 = [0.80, 1.2]**
  Limit for the amplitude wiggling offset for the second modulation frequency. Offset is higher than for the first modulation frequency and therefore limits are set between 0.8 and 1.2 (smooth limits that can be adapted for each batch of ToF modules).
- **TP.AmplitudeWigglingAmplitude1_max = [0.01,0.10]**
  Limit for the maximum amplitude for the amplitude wiggling fit for the first modulation frequency. Limit is between 0.01 and 0.10.
- **TP.AmplitudeWigglingAmplitude2_max = [0.01,0.10]**
  Limit for the maximum amplitude for the amplitude wiggling fit for the second modulation frequency. Limit is between 0.01 and 0.10.
- **TP.PhaseWigglingAmplitude1_max = [0.01,0.10]**
  Limit for the maximum amplitude for the phase wiggling fit for the first modulation frequency. Limit is between 0.01 and 0.10.
- **TP.PhaseWigglingAmplitude2_max = [0.01,0.10]**
  Limit for the maximum amplitude for the phase wiggling fit for the second modulation frequency. Limit is between 0.01 and 0.10.
- **TP.TempCompensation1 = [0.0005, 0.0015]**
  Limits for the temperature compensation for the first modulation frequency (in m/K). Depends on VCSEL and is typically around 1mm/K. Limits allow +/-0.5mm/K.
- **TP.TempCompensation2 = [0.0005, 0.0015]**
- **TP.PhaseNoiseRatio1_mean = [0.85, 1.15]**
  Limit for the ratio between phase signal and noise parameters for the first modulation frequency. As noise parameters are global settings, this test parameter limits can also be set globally to 0.85 to 1.15.
- **TP.PhaseNoiseRatio2_mean = [0.85, 1.15]**
  Limit for the ratio between phase signal and noise parameters for the first modulation frequency. As noise parameters are global settings, this test parameter limits can also be set globally to 0.85 to 1.15.
- **TP.Efficiency1_mean = [0.04, 1]**
  Limit for the mean efficiency value for the first modulation frequency (in DN*m^2/µs). Lower limit has to be adapted for the special system performance, upper limit is 100% = 1.
- **TP.Efficiency2_mean = [0.04, 1]**
  Limit for the mean efficiency value for the second modulation frequency (in DN*m^2/µs). Lower limit has to be adapted for the special system performance, upper limit is 100% = 1.
- **TP.Efficiency1_std = [0, 1]**
  Limit for the standard deviation for the efficiency value for the second modulation frequency (in DN*m^2/µs). Lower limit is 0, upper limit is 100% = 1.
- **TP.Efficiency2_std = [0, 1]**
  Limit for the standard deviation for the efficiency value for the second modulation frequency (in DN*m^2/µs). Lower limit is 0, upper limit is 100% = 1.
- **TP.DME1_mean = [0.5, 1]**
  Limit for the mean dynamic mixing efficiency value for the first modulation frequency. Lower limit is defined by 50% = 0.5, upper limit is defined by 100% = 1.

- **TP.DME2_mean = [0.5, 1]**
  Limit for the mean dynamic mixing efficiency value for the first modulation frequency. Lower limit is defined by 50% = 0.5, upper limit is defined by 100% = 1.
- **TP.DME1_std = [0.001, 0.1]**
  Limit for the standard deviation of the dynamic mixing efficiency. Lower limit is 0.001, upper limit 0.1.
- **TP.DME2_std = [0.001, 0.1]**
  Limit for the standard deviation of the dynamic mixing efficiency. Lower limit is 0.001, upper limit 0.1.
- **TP.BeamProfile1_min = [0.8, 1.3]** % beam profile central values
  The beam profile is divided into 5 equal parts (two outer parts, two middle parts and one center part) as vertical cuts through the image. It is normalized to the mean value of the inner parts. Range between lower and upper limit should be 0.5, definite values are defined by the VCSEL profile.
- **TP.BeamProfile1_max = [1.1, 1.6]** % beam profile central values
- **TP.BeamProfile2_min = [0.6, 1.1]** % beam profile middle values
- **TP.BeamProfile2_max = [0.9, 1.4]** % beam profile middle values
- **TP.BeamProfile3_min = [0.0, 1.3]** % beam profile outer values
- **TP.BeamProfile3_max = [0.3, 1.6]** % beam profile outer values
- **TP.illuminationTemperature = [18, 35]**
  Limit for the temperature of the illumination source (VCSEL) during data acquisition. Limits should be within the later OEM's use. VCSEL and other electrical components may do some further restrictions.

4. Add wiggling calibration by using LTS data

For an individual wiggling calibration data from the linear translation stage (LTS) is needed. The global wiggling parameters work as starting values for the wiggling fitting algorithm. Individual calibration can be done by loading the LTS data into the calibration script of Matlab.

5. Add individual temperature drift data

Add temperature drift data to the calibration process. The global values

TD.TempCompensation_80320000Hz = [0, 0.00113, 0, 0]

TD.TempCompensation_60240000Hz = [0, 0.00119, 0, 0]

are only used if there is no individual temperature drift available. Individual calibration can be done by loading the thermal drift data into the calibration script of Matlab.

6. Calibrate fiber box with golden sample data

For the calibration of the fiber box, the configuration file, data from the fiber box and the calibration file is needed. Be sure to use the calibration file with individual wiggling calibration (cf. section 4) and with individual temperature drift, if available (cf. section 5).

From the configuration file, the following parameters may need to be adapted for the fiber length calibration:

- **FB.pattern_size = [5, 6]**
  The number of fibers and their array: in this example that are 5 x 6 = 30 fibers. This has to be adapted for each pattern of fibers.
- **FB.SD.noise_threshold = 50**
  The threshold value between a detection of the fiber and background noise. Adaption is only necessary in case of stray light.
- **FB.intensiy_limits = [50, 1400]**
  To work properly as a fiber, the intensity value has to be between 50 and 1400. Too low or high values need to cause a change in the exposure time, not in limit changes.
- **FB.amplitude_limits = [50, 1400]**
  To work properly as a fiber, the amplitude value has to be between 50 and 1400. Too low or high values need to cause a change in the exposure time, not in limit changes.
- **FB.phase_noise_limits = [0.001, 0.05]**
  If the noise is higher than this limit, the fiber cannot be used for calibration process.
- **FB.fiber_count_limits = [21, 30]**
  The number of valid fibers that could be calibrated should be within 21 and 30. Fibers that could not be calibrated with the golden device can never be used for later calibration. In case some fibers where not calibrated properly either change threshold values or exposure times of the ToF module.

The matlab file **example_calibrate_fibers.m** from the ccalib folder works as an example for the fiber box calibration. The raw data from the fiber box is processed with the calibration values from the calibration file (FPPN is applied, wiggling and temperature drift considered, etc.) and at the end 30 fiber lengths are generated.

Those fiber lengths are then used to update the values of **FB.FiberLengths** like in the example below:

```
FB.FiberLengths = [ ...
    NaN, NaN, 1.3708, 1.2913, 1.2400, ...
    NaN, 2.0895, 1.9897, 1.8979, 1.8218, ...
    1.7587, NaN, 2.5895, 2.5221, NaN, ...
    2.3547, 2.2715, 2.1930, 3.1082, 3.0544, ...
    2.9788, 2.8807, 2.7742, NaN, NaN, …
    3.5748, 3.4974, 3.3910, NaN, NaN];
```

7. Final limit definition

Both LED box and fiber box are ready for use in mass calibration.

As a last step all limits and settings in the configuration file have to be checked finally. This also includes the fiber box limits and settings. For time reasons the debug settings should be turned off in the configuration file (**debug = 0**).

More statistical data may cause the limits to be adapted (narrowing or widening limits) and help to adjust settings for the ToF module (like exposure time).

Document History

Document title: Calibration SW instruction manual – Cal-5-2-AN

Revision	Origin of Change	Submission Date	Description of Change
0	SMa	2016-12-06	New Application Note