Calibration software – Parameter and Test points

*by **pmd**technologies*

Abstract

This document describes all parameters and test points defined in the calibration configuration file. Furthermore one finds an instruction on how to tune the calibration software parameters and set appropriate test points for new camera devices, new lenses, new illumination sources or new calibration boxes.

Prerequisites

- Calibration Basics [an-cal-2-1]
- Calibration Software [an-cal-3-4]

Table of Contents

1. Introduction

A configuration file (calibration.ini) contains various kinds of parameters and test points:

- general parameters
- box setup parameters
- calibration configuration parameters
- test points and limits

The general parameters normally have to be set only ones and should only change if bigger modifications have to be done. All other parameters need to be set individually to the pmd camera to be calibrated and some net to be adopted after first data is analyzed. The next passages will explain the individual parameters.

2. General parameters

First of all, all general parameters, have to be set. These parameters normally don't have to be changed, except there are hardware changes either on the pmd camera or on the calibration setup. The following table summarizes the general parameters. Below can be found a more detailed description of each individual parameter.

Section	Parameter	Unit	Type	Default Value	Description	Required?
Global	Config_file_version	[]	Int	1	Version of the config file layout	Yes
	debug	[]	boolean	false	generate debug plots (slows down the process)	no
	OutBaseDir	[]	string	%TEMP%	output directory for calibration results	no
	LogOutDir	[]	string	%TEMP%	output directory for logout files	no
	Reflectivity	[]	Float	0.92	Reflectivity of target	Yes
	Modulation_frequencies	[Hz]	Array of float	[80320000, 60240000]	Modulation frequencies of the pmd camera to be calibrated	Yes
	Calibration_Data_Format	[]	Int	7	Calibration content is stored in this format	No
	Calibration_Data_fpnBits	[]	Int	3	FPN bit depth of calibration content stored	No
	Calibration_Data_fppnBits	[]	Int	10	FPPN bit depth of calibration content stored	No
	Calibration_Data_Size_limit	Bytes	Int	125000	Limit of size of calibration content stored	No
	Product_code	[]	String	-	Used for camera identification in the application	Yes
	Product_identifier_string	[]	String	'PMDTOF__ ________'	Used for camera identification in the application	Yes

- **Config file version**
  The version of the configuration file layout has to be set appropriately, such that the following parameters can be read successfully into the calibration software.
- **debug**
  For configuration **debug = 1** has to be set. This creates plots during calibration to find any anomalies in the data. Also quite helpful especially at the beginning, this value should be switched to zero in mass production, due to the fact, that the image creation is quite slow.
- **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.
- **Reflectivity**
  Reflectivity of the target the pmd camera is facing at during the calibration data acquisition (LED box or LTS setup).
- **Modulation_frequencies**
  Modulation frequencies of the pmd camera, which are used during data acquisition and for which calibration content is to be generated. Values are stored as an array of integers and the unit has to be Hz. The order in the array relates to the test points, which are frequency depended. The first entry corresponds to all test points marked with _0_, the second entry corresponds to _1_ and so on.
- **Calibration_Data_Format**
  Define the output format, in which the calibration content is stored:
  Calibration_Data_Format = 7: HeaderV7 + JGF, no footer (For more information on the data format please refer to [1]).
- **Calibration_Data_FpnBits**
  The FPN values generated during calibration are compressed according to the FPN bits assigned here. Higher values allow more precise values but increase the size of the calibration data.
- **Calibration_Data_FppnBits**
  The FPPN values generated during calibration are compressed according to the FPPN bits assigned here. Higher values allow more precise values but increase the size of the calibration data.
- **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.
- **Product_code**
  The product code is necessary for identifying the pmd camera later in the application. It is therefore important to use the right product code. As this code is also related to eye safety requirements, it has to be handled very carefully.
- **Product_identifier_string**
  The product identifier string is used to show the appropriate pmd camera name to the user.

3. Calibration setup parameters

These parameters are independent of the camera settings and do not need to be changed when the camera settings are changed. However, if modifications on the boxes or on the LTS setup are performed please crosscheck that the values in the configuration file are still valid. This holds especially for the fiber length in the Fiber box section and the pattern center in the LED box section. The following table summarizes all parameters of the setups:

Section	Parameter	Unit	Type	Default Value	Description	Required?
Lens calibration	LC.pattern_mask	[]	int	-	LED pattern configuration (row- and column-wise); 1 means: IR LED should be there, 0 means no LED there; the red LED is not visible for the pmd camera (0) e.g. [1,0,1,0,1; ... 0,0,0,0,0,0]	yes
	LC.pattern_spacing	[m]	float	-	The spacing of the drill holes in the aluminum plate (typically 30mm).	yes
	LC.pattern_center	[m]	float	-	[x,y,z] vector from the lens' focal point to the mid LED (the red one)	yes, very precise!!
	LC.SpotSize	[]	int	5	spot size for the detection algorithm	no
	LC.SpotDetector_type	[]	Int	2	Definition, which spot detector algorithm is used	no
	LC.SD.noise_threshold	[DN]	int	100	Noise threshold for spot detection	Yes
	LC.lens_model	[]	int	1	1: polynomial model, 2: fish-eye model (not yet supported)	No
	LC.lens_parameter	[]	float	-	initial estimation of the lens parameters (only used for the detection and the fit's initial values) [fx,fy,cx,cy,k1,k2,p1,p2,k3]	yes
Fiber box (wiggling)	FB.FiberLenghts	[m]	array of floats	-	calibrated fiber lengths (one-time calibration of fiber box itself using golden sample required)	yes, for fiber box based calibration
	FB.pattern_size	[]	int	[5,6]	row * colums fiber spots	yes, for fiber box based calibration
	FB.SD.noise_threshold	[DN]	int	50	Noise threshold for spot detection	yes, for fiber box based calibration
	FD.intensity_limits	[DN]	Array of int	[50, 1400]	Pixels with values out of the limits will be ignored	yes, for fiber box based calibration
	FB.amplitude_limits	[DN]	Array of int	[50, 1400]	Pixels with values out of the limits will be ignored	yes, for fiber box based calibration
	FB.phase_noise_limits	[rad]	Array of floats	[0.001,0.05]	Pixels with values outside of the limits will be ignored	yes, for fiber box based calibration
	FB.fiber_count_limits	[]	Array of int	[21,30]	Number of Fibers, which have to be detected for proper wiggling calibration	yes, for fiber box based calibration
	FB.AmplitudeWigglingFit_MSE_limits	[]	Array of floats	[0, 0.001]	Limits of mean spared error of amplitude wiggling fit	yes, for fiber box based calibration
	FB.PhaseWigglingFit_MSE_limits	[]	Array of floats	[0, 0.0005]	Limits of mean spared error of phase wiggling fit	yes, for fiber box based calibration
LTS calibration	LTS.amplitude_limits_center	[DN]	Array of int	[20, 1400]	For center detection only distances with amplitude values within limits are used	Yes

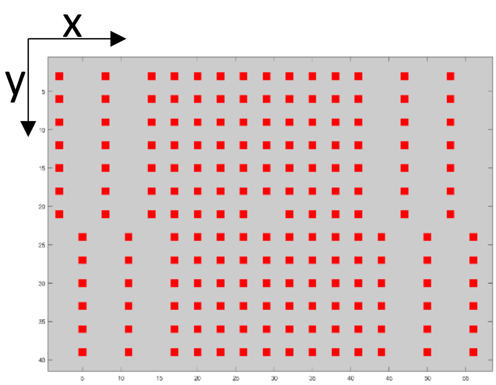
	LTS.amplitude_limits	[DN]	Array of int	[5, 1400]	Only distances with valid amplitudes are used for wiggling compensation	Yes
	LTS.valid_frames_limit	[]	Int	15	Minimum limit for valid frames at one distance for wiggling compensation	Yes
Stray light parameters	Slp.xxx	[]	-	-	Global stray light parameters to be stored in the calibration content	no
Wiggling parameters	AmplitudeWiggling_0.xxx	[]	-	-	Amplitude wiggling parameters for first modulation frequency, start values or global	Yes
	AmplitudeWiggling_1.xxx	[]	-	-	Amplitude wiggling parameters for second modulation frequency, start values or global	Yes
	PhaseWiggling_0.xxx	[]	-	-	Phase wiggling parameters for first modulation frequency, start values or global	Yes
	PhaseWiggling_1.xxx	[]	-	-	Phase wiggling parameters for second modulation frequency, start values or global	Yes
Noise Parameters	NP.NoiseParameter_0	[]	Array of float	-	noise parameters for the first modulation frequency, start values or global	Yes
	NP.NoiseParameter_1	[]	Array of float	-	noise parameters for the second modulation frequency, start values or global	Yes
Temperature Drift	TD.TempCompensation_0	[]	Array of float	-	temperature drift parameters for the first modulation frequency, start values or global	Yes
	TD.TempCompensation_1	[]	Array of float	-	temperature drift parameters for the second modulation frequency, start values or global	Yes
	TD.amplitude_limits	[DN]	Array of float	[50,1400]	Only pixels with amplitude values within limits are used	no
	TD.temperature_jump_limit	[]	Float	0.5	Illumination temperature between individual datasets should not change more than this limit	no
Defect Pixel / Mask	DP.Inclination_angle	[degree]	float	-	if indicated, pixels outside this maximum inclination angle will be masked out	no
	DP.DarkCurrent	[DN/s]	Array of int	[-10, 10]	Pixels with dark current values outside limits will masked as defect pixels	Yes
	DP.FPN	[DN]	Array of int	2047+[-200,200]	Pixels with FPN values outside limits will masked as defect pixels	Yes
	DP.DME_0	[]	Array of float	[0.5, 1.1]	Pixels with DME values outside limits will masked as defect pixels, here for the first modulation frequency	Yes
	DP.DME_1	[]	Array of float	[0.5, 1.1]	Pixels with DME values outside limits will masked as defect pixels, here for the second modulation frequency	Yes
	DP.FPPN_0	[m]	Array of float	[-10, 10]	Pixels with FPPN values outside limits will masked as defect pixels, here for the second modulation frequency	Yes
	DP.FPPN _1	m]	Array of float	[-10, 10]	Pixels with FPPN values outside limits will masked as defect pixels, here for the second modulation frequency	Yes

	DP.Amplitude_0	[DN]	Array of int	[30, 2500]	Pixels with amplitude values outside limits will masked as defect pixels, here for the second modulation frequency	Yes
	DP. Amplitude _1	[DN]	Array of int	[30, 2500]	Pixels withamplitude values outside limits will masked as defect pixels, here for the second modulation frequency	Yes
	DP.Intensity	[DN]	Array of int	[10, 2500]	Pixels with intensity values outside limits will masked as defect pixels, here for the second modulation frequency	Yes

3.1. LED Box

- **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. The pattern needs to be adopted according to the used lens. 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; ...
  1,0,1,0,1,1,1,1,0,1,1,1,1,1,0,1,0,1,0; ...
  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 = 0.03**.
  The distance between two drill holes in the grid is currently 3cm.
- **LC.pattern_center = [9*0.03, 6*0.03, 0.1758]**
  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). Once the LED box is set up, the most critical part is the third coordinate, as this one can easily change due to a different tray due to module changes. The third coordinate is also very important for other parameters than the lens calibration and therefore has to be set correctly.
- **LC.SpotSize = 5**
  The size of the LED spots to be detected. Standard value is **5** and normally does not need to be adapted.

- **LC.SpotDetector_type = 2**
  The detector type can be set to **1** (spot detection) or **2** (threshold detection). Standard detector type is **2** and normally does not need to be adapted.
- **LC.SD.noise_threshold = 100**
  The threshold value between a spot and background noise. The standard value only needs to be adapted if new kind of LEDs are used that differ from older ones concerning the properties.
- **LC.lens_model = 1**
  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. Please see section 4 for these test points. fx and fy can be taken from the data sheet, cx and cy should be in the center of the ROI.

3.2. Fiber box

The fiber box parameters are only needed, if the calibration setup includes a fiber box. For more information on the setup options please refer to [2]. If used, the following parameters need to be set:

- **FB.FiberLengths**
  For each fiber the length has to be determined in a special procedure described in [3]. These determined fiber length need to be implemented here to be used for all other pmd cameras. An example looks like:

  ```
  FB.FiberLengths = [1.2308, 1.2995, 1.3886, 1.4663, ...
          1.5626, 1.6568, 1.7370, 1.8105, 1.8995, 1.9752, ...
          2.0747, 2.1428, 2.2452, 2.3277, 2.4091, 2.5023, ...
          2.5727, 2.6746, 2.7524, 2.8422, 2.9119, 2.9789, ...
          3.0712, 3.1694, 3.2490, 3.3716, 3.4515, 3.5289, ...
          3.6228, 3.6995];
  ```

- **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.
- **FB.AmplitudeWigglingFit_MSE_limits = [0, 0.001]**
  Limits to check if the wiggling fit for the amplitude wiggling worked well. This gives the allowed range of the mean squared error between measurement and fitting function.
- **FB.PhaseWigglingFit_MSE_limits = [0, 0.0005]**
  Limits to check if the wiggling fit for the phase wiggling worked well. This gives the allowed range of the mean squared error between measurement and fitting function.

3.3. Linear translation stage

The linear translation stage parameters are only needed, if a linear translation stage is used for determining for example the wiggling parameters. Please refer to [2] for more details on the setup options. The following parameters are to be set:

- **LTS.amplitude_limits_center = [20, 1400]**
  In order to use the LTS data for calibration, the center needs to be determined, which is the pixel, which is looking perpendicular to the wall. Only amplitude values within the range are used to determine this center.
- **LTS.amplitude_limits = [5, 1400]**
  For data evaluation only amplitude values within the range set here are used to get the calibration values. If too many values are flagged out, please take a look at the used exposure times and adapt these accordingly.
- **LTS.valid_frames_limit = 15**
  Due to too high noise sometimes only a few frames are useful for determining the calibration values. This limit gives the minimum number of valid frames, which is required for calibration.

3.4. Stray light parameters

If stray light is to be corrected, here global parameters can be set. These parameters have to be determined with a method described in [4]. Currently, the software will just implement the values set here into the calibration file.

3.5. Wiggling parameters

The wiggling parameters are divided into two categories, one for the amplitude wiggling and one for the phase wiggling. Each of these sections is further divided for each modulation frequency used. The parameters are set global or used as starting parameters to determine the actual correction function due to a fitting algorithm. The currently implemented underlying function is a superposition of different sinusoidal functions and thus parameters for offset, harmonic, amplitude and phase have to be set. An example for the amplitude wiggling of the first modulation frequency (indicated by _0) looks like:

AmplitudeWiggling_0.Offset = 0.6907;

AmplitudeWiggling_0.Harmonic = [2, 4];

AmplitudeWiggling_0.Amplitude = [0.0038, 0.0439];

AmplitudeWiggling_0.Phase = [3.8003, 4.0405];

3.6. Noise parameters

For each modulation frequency the noise parameters are to be set in this section. These parameters are either global or will be used as starting parameters for fitting the individual calibration values. The description of the different values can be found in [2]. An example for the first modulation frequency (indicated by _0) looks like:

NP.NoiseParameter_0 = [0.0176, 4.22, 0.0122, 23.2]

3.7. Temperature drift

Temperature drift coefficients can be either set global or as starting parameters for fitting the actual parameters for each module for each modulation frequency. If temperature drift raw data is found new parameters will be determined otherwise, the values written here will be implement into the calibration file. The second value refers to the slope of a linear fit function and is currently only supported, the other parameters are not used and set to 0. For more information please refer to [2]. An example for the first modulation frequency (indicated by _0) looks like:

TD.temperature_compensation_0 = [0, 0.00113, 0, 0]

If temperature drift raw data is present, then the following to criteria will be checked:

- **TD.amplitude_limits = [50,1400]**
  Only amplitude values within the limits are used to determine new temperature compensation parameters. The upper limit is defined due to avoiding possible saturation and the lower limit is for enough signal detection. If too many pixels are flagged out, the exposure time should be adapted appropriately.
- **TD.temperature_jump_limit = 0.5**
  If temperature jumps higher than the here set limit are detected a warning will be placed to check if the temperature determination looks useful. If this limit is reached, a closer look at the raw data is necessary to see if the linear fitting function is still appropriate.

3.8. Defect pixel / mask generation

Defect pixels can be found due to several reasons. If a pixel is found as a defect pixel according to the following parameters, this pixel will be masked and not used later in the application. In addition to the defect pixels it is possible to create a mask by hand by setting the inclination angle according to a specification for example. Thus the mask is a combination of defects and hand set values. The actual field of view in horizontal and vertical direction then can be tracked by the last two parameters (HFoV, VFoV):

- **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.
- **DP.HFoV**
  Determines the field of view from the center to the first/last not masked pixel in horizontal direction. Thus values between [0, 90] degrees can be determined and will be tracked.
- **DP.VFoV**
  Determines the field of view from the center to the first/last not masked pixel in vertical direction. Thus values between [0, 90] degrees can be determined and will be tracked.

4. Calibration test points

At the end of the calibration procedure the calculated calibration values are checked and several tests points need to be fulfilled before the calibration is stated as successful. The following table lists all currently implemented test points:

Section	Parameter	Unit	Type	Hint	Description
Lens calibration	LC.pixel_mapping_limits	[Pixels]	Array of int	[0,10]	For spot identification, pixels are identified within limits
	LC.spot_count_limits	[]	Array of int	[80,134]	Number of LED spots, which have to be found
	LC.fx_limits	[]	float	<+-5%	limits for fx
	LC.fy_limits	[]	float	<+-5%	limits for fy
	LC.cx_limits	[]	float	<+-5%	limits for cx
	LC.cy_limits	[]	float	<+-5%	limits for cy
	LC.projection_error_limits	[pixel]	float	<0.3	limits of the fit quality (the projection error)
	LC.rvec_x_limits	[rad]	float	<0.1	Limits for rotation toleration during calibration
	LC.rvec_y_limits	[rad]	float	<0.1	Limits for rotation toleration during calibration
	LC.rvec_z_limits	[rad]	float	<0.1	Limits for rotation toleration during calibration
Global	TP.valid_pixel_count	[]	int	ROI-specific	after mask application, this number of pixels should remain
	TP_defect_pixel_count	[]	Int	[0, 40]	Maximum allowed number of defect pixels without masked by hand pixels
	TP.defect_cluster_size	[]	Int	[0, 2]	maximum allowed defect cluster size
	TP.FPN_mean	[ADC units]	int	[2000..2100]	mean of FPN must be within this range
	TP.DarkCurrent_mean	[]	float	[-0.1..0.1]	mean of dark current
FPPN	TP.PhaseSTD_0_mean	[m]		[1-e4..2e-2]	mean of the phase of the FPPN acquisition for first modulation frequency
	TP.PhaseSTD_1_mean	[m]		[1-e4..2e-2]	mean of the phase of the FPPN acquisition for second modulation frequency
	TP.PhaseSTD_0_max	[m]		[0..0.1]	max of the phase of the FPPN acquisition for first modulation frequency
	TP.PhaseSTD_1_max	[m]		[0..0.1]	max of the phase of the FPPN acquisition for second modulation frequency
	TP.Amplitude_0_mean	[]		[400 .. 1400]	mean of the amplitude of the FPPN acquisition for the first modulation frequency
	TP.Amplitude_1_mean	[]		[400 .. 1400]	mean of the amplitude of the FPPN acquisition for the second modulation frequency
	TP.Amplitude_0_max	[]		[400 .. 1400]	max amplitude of the FPPN acquisition for the first modulation frequency
	TP.Amplitude_1_max	[]		[400 .. 1400]	max amplitude of the FPPN acquisition for the second modulation frequency
	TP.FPPN_0_std	[m]	Float	[0, 0.01]	Maximum allowed fppn std values for first modulation frequency
	TP.FPPN_1_std	[m]	Float	[0, 0.01]	Maximum allowed fppn std values for second modulation frequency
Wiggling	TP.AmplitudeWigglingOffset_0	[]	float	[0.6..1.2]	Offset range of amplitude wiggling of fitting function for the first modulation frequency
	TP.AmplitudeWigglingOffset_1	[]	float	[0.6..1.2]	Offset range of amplitude wiggling of fitting function for the second modulation frequency
	TP.AmplitudeWigglingAmplitude_0_max	[]	float	[0.01..0.1]	amplitude range of amplitude wiggling of fitting function for the first modulation frequency
	TP.AmplitudeWigglingAmplitude_1_max	[]	float	[0.01..0.1]	amplitude range of amplitude wiggling of fitting function for the second modulation frequency
	TP.PhaseWigglingAmplitude_0_max	[]	float	[0.01..0.1]	Amplitude range of phase wiggling of fitting function for the first modulation frequency
	TP.PhaseWigglingAmplitude_1_max	[]	float	[0.01..0.1]	Amplitude range of phase wiggling of fitting function for the second modulation frequency

Noise parameters	TP.PhaseNoiseRatio_0_mean	[]	float	[0.8..1.2]	verify noise parameters for the first modulation frequency
	TP.PhaseNoiseRatio_1_mean	[]	float	[0.8..1.2]	verify noise parameters for the second modulation frequency
Temperature drift	TP.TempCompensation_0	[]	float	<0.001	lower / upper limit of the temp. compensation slope
	TP.TempCompensation_1	[]	float	<0.001	lower / upper limit of the temp. compensation slope
Efficiency	TP.Efficiency_0_mean	[(DN*m^2)/µs]	float	[0.17, 1]	Mean efficiency for first modulation frequency
	TP.Efficiency_1_mean	[(DN*m^2)/µs]	float	[0.17, 1]	Mean efficiency for second modulation frequency
	TP.Efficiency_0_std	[(DN*m^2)/µs]	float	[0, 1]	Efficiency noise for first modulation frequency
	TP.Efficiency_1_std	[(DN*m^2)/µs]	float	[0, 1]	Efficiency noise for second modulation frequency
DME	TP.DME_0_mean	[]	float	[0.5, 1]	Mean dynamic mixing efficiency for first modulation frequency
	TP.DME_1_mean	[]	float	[0.5, 1]	Mean dynamic mixing efficiency for second modulation frequency
	TP.DME_0_std	[]	float	[0.001, 0.1]	Dynamic mixing efficiency noise for first modulation frequency
	TP.DME_1_std	[]	float	[0.001, 0.1]	Dynamic mixing efficiency noise for second modulation frequency
Beam profile	TP.BeamProfile1_min	[]	float	[0.8, 1.3]	Minimum value for the beam profile at the center
	TP.BeamProfile1_max	[]	Float	[0.8, 1.3]	Maximum value for the beam profile at the center
	TP.BeamProfile2_min	[]	Float	[0.7, 1.2]	Minimum value for the beam profile at the middle parts
	TP.BeamProfile2_max	[]	Float	[0.8, 1.4]	Maximum value for the beam profile at the middle parts
	TP.BeamProfile3_min	[]	Float	[0.0, 1.3]	Minimum value for the beam profile at the outer parts
	TP.BeamProfile3_max	[]	float	[0.0, 1.6]	Maximum value for the beam profile at the outer parts
Illumination Temperature	TP.illuminationTemperature	[°C]	Int	[18, 35]	Illumination temperature during calibration.

4.1. Lens calibration

- **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. Only needed for first spot identification. Ones the LED spots are found, this has no impact on the calibration results.
- **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).

4.2. Global limits

- **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. Limits 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

4.3. FPPN limits

- **TP.PhaseSTD_0_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.PhaseSTD_1_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.PhaseSTD_0_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.PhaseSTD_1_max = [0, 1.25]**
  Limit for the maximum phase standard deviation of the second modulation frequency (in m). Lower limit is 0. Upper limit is defined by the spec.
- **TP.Amplitude_0_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.Amplitude_1_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.Amplitude_0_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.Amplitude_1_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.FPPN_0_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.FPPN_1_std = [0, 0.1]**
  Limit for the standard deviation of the FPPN for the second modulation frequency. Lower limit is defined by 0 (no standard deviation), upper limit at 0.1m.

4.4. Wiggling limits

- **TP.AmplitudeWigglingOffset_0 = [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.AmplitudeWigglingOffset_1 = [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.AmplitudeWigglingAmplitude_0_max = [0.01,0.10]**
  Limit for the maximum amplitude of all harmonics for the amplitude wiggling fit for the first modulation frequency. Limit is between 0.01 and 0.10.
- **TP.AmplitudeWigglingAmplitude_1_max = [0.01,0.10]**
  Limit for the maximum amplitude of all harmonics for the amplitude wiggling fit for the second modulation frequency. Limit is between 0.01 and 0.10.
- **TP.PhaseWigglingAmplitude_0_max = [0.01,0.10]**
  Limit for the maximum amplitude of all harmonics for the phase wiggling fit for the first modulation frequency. Limit is between 0.01 and 0.10.
- **TP.PhaseWigglingAmplitude_1_max = [0.01,0.10]**
  Limit for the maximum amplitude of all harmonics for the phase wiggling fit for the second modulation frequency. Limit is between 0.01 and 0.10.

4.5. Noise parameter limits

- **TP.PhaseNoiseRatio_0_mean = [0.85, 1.15]**
  Limit for the ratio between phase signal and noise parameters for the first modulation frequency. As noise parameters are mostly global settings, this test parameter limits can also be set globally to 0.85 to 1.15.
- **TP.PhaseNoiseRatio_1_mean = [0.85, 1.15]**
  Limit for the ratio between phase signal and noise parameters for the second modulation frequency. As noise parameters are mostly global settings, this test parameter limits can also be set globally to 0.85 to 1.15.

4.6. Temperature drift limits

- **TP.TempCompensation_0 = [0.0005, 0.0015]**
  Limits for the temperature compensation for the first modulation frequency (in m/K). Depends on the illumination part and is typically around 1mm/K. Limits allow +/-0.5mm/K.
- **TP.TempCompensation_1 = [0.0005, 0.0015]**
  Limits for the temperature compensation for the second modulation frequency (in m/K). Depends on the illumination part and is typically around 1mm/K. Limits allow +/-0.5mm/K.

4.7. Efficiency limits

- **TP.Efficiency_0_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 according to the specification, upper limit is 100% = 1.
- **TP.Efficiency_1_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 according to the specification, upper limit is 100% = 1.
- **TP.Efficiency_0_std = [0, 1]**
  Limit for the standard deviation for the efficiency value for the first modulation frequency (in DN*m^2/µs). Lower limit is 0. Upper limit needs to be defined according to the camera performance.
- **TP.Efficiency_1_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 needs to be defined according to the camera performance.

4.8. DME limits

- **TP.DME_0_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.DME_1_mean = [0.5, 1]**
  Limit for the mean dynamic mixing efficiency value for the second modulation frequency. Lower limit is defined by 50% = 0.5, upper limit is defined by 100% = 1.
- **TP.DME_0_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.DME_1_std = [0.001, 0.1]**
  Limit for the standard deviation of the dynamic mixing efficiency. Lower limit is 0.001, upper limit 0.1.

4.9. Beam profile limits

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 middle parts. Values need to be defined according to the illumination beam profile specification.

- **TP.BeamProfile1_min = [0.8, 1.3]**
  Limits for the beam profile minimum for the center part.
- **TP.BeamProfile1_max = [1.1, 1.6]**
- Limits for the beam profile maximum for the center part.
  TP.BeamProfile2_min = [0.6, 1.1]
  Limits for the beam profile minimum for the middle parts.
- **TP.BeamProfile2_max = [0.9, 1.4]**
  Limits for the beam profile maximum for the middle parts.
- **TP.BeamProfile3_min = [0.0, 1.3]**
- Limits for the beam profile minimum for the outer parts.
  TP.BeamProfile3_max = [0.3, 1.6]
  Limits for the beam profile maximum for the outer parts.

4.10. Illumination temperature

- **TP.illuminationTemperature = [18, 35]**
  Limit for the temperature of the illumination source during data acquisition. Limits should be within the later OEM's use. VCSEL and other electrical components may do some further restrictions.

5. Parameter and limit setup and adjustment

Most of the parameters are to be set according to the equipment used. Of course, different boxes need to have different parameters. In order to define limits, the specification of the later device should be taken into account. According to this specification several limits should be set, for example the efficiency limits. However, some limits are based on statistics. For these limits initial cameras need to be placed into the boxes and a calibration run is started. The first calibration parameters and limits are global limits and thus the calibration will most probably fail. This is no issue as these parameters are some first starting points and according to the observed results will be changed. Then a new run is taken and it is checked if the test points are now fulfilled. Once more statistical data is available, the parameters need to be tuned further, such that a stable and useful calibration is setup in the end, which is ready for mass production.

In order to identify the limits, which are to be adapted, so called log files are generated during the calibration process. These log files include all test points and thus show, at which test points values need to be changed. These log files have the following structure and can be opened with a simple text editor:

Parameter	Example	Description
TimeStamp	1515064148	Timestamp, which indicates, when the data was created
SensorSerial	0000-0000-0000-0000	Sensor serial of the calibrated camera
TestPointNumber	0005006	Each individual test point has an internal unique well defined number
TestPointName	Cy	Test point name as given in the calibration .ini file
Unit	[px]	Unit of the test point
MeasuredValue	85.273	Value calculated from the calibration script
LowerLimit	77	Lower limit from the calibration .ini file for this test point
UpperLimit	95	Upper limit from the calibration .ini file for this test point
Pass/Fail	1	Result for this test point (pass = 1, fail = 0)

The different parameters are separated by a semicolon and for each test point a new row is used. The last row of the log file gives the overall result with a PASS/FAIL conclusion.

Roles and responsibilities

pmdtechnologies does support in the design process and provides consulting support in form of design and specification reviews. pmdtechnologies is not responsible for the final product of the customer. Reference designs are recommendations and may not be suitable for a claim of completeness.

References

[1] pmd technologies ag. Calibration Data Content [an-cal-4-5] (2017).

[2] pmd technologies ag. Calibration Basics [an-cal-2-1] (2017).

[3] pmdtechnologies ag. Fiber Box Concept [an-cal-3-2] (2018).

[4] pmdtechnologies ag. Stray Light Calibration [an-cal-3-6] (2017).

Document History

Document title: Calibration Software Parameter and Test Points [an-cal-4-4]

Revision	Origin of Change	Submission Date	Description of Change
0	MRe	2019-01-30	New Application Note