Customer Documentation pmd3_data acquisition SW

*by **pmd**technologies*

Abstract

This document describes the data acquisition software used for the calibration. The configuration and the execution of the program will be explained.

Table of Contents

List of Figures

1. Introduction

The calibration is split into the data acquisition and the data processing part. The standard workflow consists of storing the data to disk and then processing this data offline. While it is easy for a high-volume production to obviate the relatively slow data storage part, it simplifies debugging and ramping up a lot.

In this documentation, the data acquisition part is explained. This comprises the configuration and the execution.

The acquisition is subdivided into measurement phases. Each phase acquires the data for one dedicated calibration stage. So, for instance, there is a measurement phase called 'FPPN' etc.

1.1. Acquisition Configuration

The configuration is subdivided into a global section and a phase-specific section. The global parameters are set once during program / camera initialization and are valid for the whole program execution. See appendix for an example configuration.

Parameter	Unit	Type	Description	Default Value	Required?
OutBaseDir	[]	directory	This is the base directory. The final directory is built as follows: <OutBaseDir>_<ModuleID>_<OutBaseSuffix>	%TEMP%	no
OutBaseSuffix	[]	string	See <OutBaseDir>	empty	no
Box_LEDs_ComPort	[]	string	The COM port to which the Arduino (LED control interface) is connected to.	The program tries to identify the COM port automatically by looking for the PID and VID of the USB device (assuming that a virtual com port is used).	if the auto detection fails; only for LED box
Use_Isel	[]	boolean	use a Isel LTS	false	only for a LTS system based on an Isel stage
Use_Compax3	[]	boolean	use a Compax3 LTS	false	only for a LTS system based on an Isel stage
UseMoveSim	[]	boolean	use a simulated LTS - useful for debugging purposes	false	only for debugging purposes
Isel_xoffset	[m]	float	When you move the LTS to position X, the script assumes that the distance from the camera module to the target amounts to X. If there is an offset (e.g. due to the module fixture), this offset goes here.	0	only for a LTS system based on an Isel stage
Isel_homeposition	[m]	float	This is where the LTS is moved after finishing data acquisition	-	only for a LTS system based on an Isel stage
UEYE_Acquire	[]	boolean	boolean value; if true, an IDS µEye camera is used to measure the beam profile	false	only for a box with µEye imager
UEYE_pmd_tint	[µs]	uint	integration time for the pmd sensor for the µEye acquisition	1000	only for a box with µEye imager
SavePhasePreview	[]	boolean	if true, the preview images are saved to disk (cf. 'Phase_Preview' parameter)	false	no
Phase_Preview	[]	boolean	if true, shows a preview image of amplitude / phase / intensity before storing the data	false	no

Table 1: Global configuration parameters

Apart from these global parameters, each measurement phase needs to be configured individually. This configuration follows the rules of a very simple state machine. This means that the parameters are configured line-by-line, and a measurement phase is not stored (alias activated) unless it is ‘committed’. After committing a phase, the phase parameters are not reset!

Parameter	Unit	Type	Description	Default Value	Required?
Phase_Tint	[µs]	uint	Integration time for this specific phase. Vector of ten entries.	(depends on operation mode)	yes
Phase_Duration	[s]	uint	Duration of the phase in seconds. Zero means 'as fast as possible'. Useful especially for temperature drift measurement. The number of frames configured will be stored within the time interval specified by this parameter.	0	yes, for temperature drift measurement
Phase_NFrames	[]	uint	Number of frames to store. Useful for averaging purposes, i.e. dealing with noise.	-	yes
Phase_LEDs	[]	string	If true, the IR LEDs in the LED-calibration box will be turned on via the Arduino micro controller. "False" will be interpreted as "Off", "true" as "On".	-	yes, for the LED box
Phase_OperationMode	[]	string	Designation of the operation mode. Cf. Royale documentation.	-	yes
Phase_commit	[]	string	activates the phase	*not applicable*	yes
Phase_Position	[m]	float	Position to which the LTS shall before prior to acquiring data	-	yes, for LTS-based systems
Phase_CalcPositions	[m]	float	Position range and interval for data acquisition. Example: "0.3 4.7 0.1" means: starting from 0.3m, move the LTS in steps of 0.1m and acquire the data at each position. Stop at 4.7m. The three float values have to be separated by a single space character.	-	yes, for LTS-based systems

Table 2: Phase specific configuration parameters

Note on choosing the integration time: You will want to choose a large integration time for calibration in order to minimize the influence from noise while at the same time avoiding over-saturation. Modulation amplitudes exceeding 1400 digits can be considered as over-exposed.

1.2. Program Execution / Computer Prerequisites

There are two versions of the program. The standard version is implemented in Matlab 2015b (pmd3.m), and then there is Python version (pmd3py.py) with the same functionality. The latter allows you to renounce for a Matlab license for the data acquisition station. If you use the same computer for the data acquisition and the actual calibration (which is Matlab-only), use the Matlab version of the data acquisition.

The program assumes that the configuration files are stored in the same folder as ‘pmd3.m’ (Matlab version) / ‘pmd3py.py’ (Python version). The file extension must be ‘.ini’.

Both implementations assume a Windows 7 64-bit computer. Of course, the underlying pmd Royale API is available for 32 bit and for Linux / Mac, too, but right now, the calibration scripts use only this very specific version of Royale.

To start, simply call ‘pmd3’ from the Matlab prompt with the working directory being the calibration root directory. Ensure that all sub-directories are within the path (‘addpath(genpath(pwd))’). Python: run pmd3py.py. Ensure that pyserial, Tk, matplotlib and numpy packages are installed.

1.3. Program Structure

The program is kept relatively simple. As shown in Figure 1, it has three interfaces. The interface to the camera is a Matlab / Python wrapper to the pmd Royale API (a separate documentation is available for this). Then there is a simple serial interface to an Arduino Micro controller, which is responsible for switching on and off the LEDs in the LED box. Furthermore, there is an interface to a linear translation stage. At pmd, we use a stage from the manufacturer Isel, and the interface is a simple serial communication. You have to customize it for your specific stage. However, look at the class and make sure that you derive from the same base class in order to have compatible interfaces.

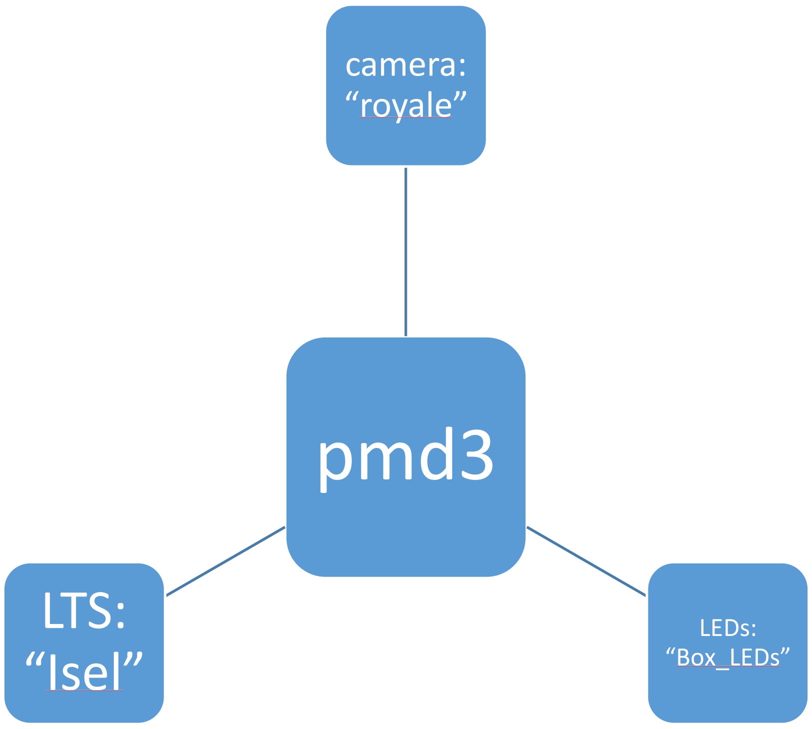

Figure 1: Program structure (interfaces)

1.4. Program Output

The output of the program consists of multiple RRF files (Royale recording files). The configuration file is also copied to the output directory. In the Matlab version, the main script (execute.m) is also copied to the output directory. Both the configuration file and the main script are useful for debugging purposes.

2. Appendix: Example Configuration Files

2.1. Example configuration file for the LED box:

```
OutBaseDir 'D:/Projekte'
OutBaseSuffix 'LED'
Box_LEDs_ComPort 'auto'

##########################################################
#        Temperature drift                               #
##########################################################
Phase_Duration 60
Phase_OperationMode 'MODE_9_10FPS_1000'
Phase_Tint 1000*ones(1,9)
Phase_NFrames 10
Phase_LEDs 'false'
Phase_commit 'TemperatureDrift'

##########################################################
```

```
#        Optic Model                                        #
#############################################################
Phase_OperationMode 'MODE_10_5FPS_2000'
Phase_Duration 0
Phase_NFrames 20
Phase_LEDs 'true'
Phase_Tint 800*ones(1,10)
Phase_commit 'OpticModel_500'

#############################################################
#        FPPN                                               #
#############################################################
Phase_LEDs 'false'
Phase_commit 'FPPN_500'

#############################################################
#        FPN                                                #
#############################################################
Phase_Tint ones(1,10)
Phase_commit 'FPN'
```

2.2. Example configuration file for the fiber box:

```
OutBaseDir 'D:\Projekte\'
OutBaseSuffix 'Fiber'

#############################################################
#        Temperature drift                                  #
#############################################################
Phase_Duration 30
Phase_OperationMode 'MODE_9_10FPS_1000'
Phase_NFrames 5
Phase_commit 'TemperatureDrift'

#############################################################
#        fiber                                              #
#############################################################
Phase_Duration 0
Phase_OperationMode 'MODE_10_5FPS_2000'
Phase_Tint 500*ones(1,10)
Phase_NFrames 20
Phase_commit 'Fiber'
```

2.3. Example configuration file for a LTS system:

```
OutBaseDir 'D:/Projekte'

OutBaseSuffix 'LTS'

UseIsel 1
Isel_xoffset 0.085
Isel_homeposition 3.5
Isel_swing_delay 2
```

```
##########################################################
#        Temperature drift                               #
##########################################################
Phase_Duration 60
Phase_OperationMode 'MODE_9_10FPS_1000'
Phase_Tint 1000*ones(1,9)
Phase_NFrames 10
Phase_Position 0.5
Phase_LEDs 'false'
Phase_commit 'TemperatureDrift'

##########################################################
#        LTS                                             #
##########################################################
Phase_OperationMode 'MODE_10_5FPS_2000'
Phase_Duration 0
Phase_NFrames 20
Phase_CalcPositions 0.10 4.7 0.1
Phase_Tint 800*ones(1,10)
Phase_commit 'LTS'

##########################################################
#        FPPN                                            #
##########################################################
Phase_Position 0.5
Phase_commit 'FPPN_500'

##########################################################
#        FPN                                             #
##########################################################
Phase_Tint ones(1,10)
Phase_Position 0.5
Phase_commit 'FPN'
```

Literature

1. R. Lange, "3D Time-of-Flight distance measurement with custom solid-state image Sensors in CMOS/CCD-technology" Siegen, 2000.

Document History

Document title: pmd3_data acquisition SW – Cal-2-2-AN

Revision	Origin of Change	Submission Date	Description of Change
0	OLo	2016-02-19	New Application Note