Time-of-Flight Validation Test Points (Quantum and Validate_Plane)

*by **pmd**technologies*

Abstract

This document introduces the two different validation tools Quantum (device level) and Validate_Plane (module level) and provides a detailed description of all validation test points that are available in both validation tools. This description does not cover validation test points that are available in only one validation tool, either Quantum or Validate_Plane.

Prerequisites

- ToF Basics [an-g-2-1]

Table of Contents

List of Figures

1. Introduction

During the production process of a Time-of-Flight (ToF) camera device it is recommended to verify the quality of the ToF system, i.e. hardware and calibration, at least once. Typically, this verification is performed twice: after module production as an outgoing quality check (OQC) on ODM side (module level validation) and after device assembly (device level validation) on OEM side. The ToF validation can additionaly be used as an incoming quality check (IQC) for the ToF modules on OEM side.

The Validation Box [1] was designed to allow for a fast and robust validation of a ToF module or device. Validation software algorithms are provided by the Validate_Plane and Quantum software tool. These algorithms yield several numerical values that are supposed to be suitable for testing predefined performance critera as well as to identify hardware failures. A customer may modify Validate_Plane or Quantum to create an own ToF validation application and apply test limits for PASS/FAIL analysis.

The current implementation of the validation process in Validate_Plane and Quantum is based on the evaluation of

- 20 amplitude images and
- 20 point clouds.

Typically, these amplitude images and point clouds are calculated from the same 20 raw frames. All frames are acquired in a Validation Box environment with a structureless, diffusive reflectiving planar target at a fixed distance (e.g. 50 cm). In order to be able to identify an insufficient or imperfect calibration, it is important that this distance is chosen in a way that it differs from the size of the LED Box [2]. The setup is designed so that all calibrated time-of-flight pixels should be able to record non-biased measurement data for evaluation.

2. Validation Tools

As already mentioned before, two different validation tools can be provided by pmd: Validate_Plane to be used on module level and Quantum for the validation on device level.

2.1. Validate_Plane

Validate_Plane is a Python based software tool that allows for ToF validation on module level. It requires 20 frames of recorded raw data in rds-file format as input. In a first step, this raw data is processed using the Spectre processing framework, resulting in 20 amplitude images and 20 point clouds. In a second step, these processed data is then evaluated and the numerical results of the various validation test points are calculated. In a last step, the calculation results are stored in xlsx-file format.

Note: The current implementation of Validate_Plane does not perform a comparison of the validation test points with predefined limits. Thus, it does not result in a PASS/FAIL decision. In case a PASS/FAIL decision is required, the xlsx-output can be further analyzed using a standalone software tool.

2.2. Quantum

Quantum can be understood as the counterpart of Validate_Plane for ToF validation on device level. Quantum is a Java application for Android running on the final device. Different to Validate_Plane it does not require previously recorded raw data as input. When running Quantum, it will subsequently record all necessary raw data, process this raw data to yield the

20 amplitude images and 20 point clouds and calculate the numerical results of the validation test points. In addition, Quantum compares these values to predefined limits, thus finally resulting in a PASS/FAIL decision. The numerical results of the ToF validation are stored in csv-file format on the device.

3. Validation Test Points

The following subsections provide a detailed description of all the validation test points that are available in both validation tools, Validate_Plane and Quantum.

3.1. Spatial Depth Precision (depth_precision_spatial)

The spatial depth precision (test point "depth_precision_spatial") is a measure of the spatial noise of the depth values. It is given by the standard deviation of all depth values of a single frame. In order to compensate for a possible tilting of the module/device, a plane fit is performed for this single frame and the depth values of that plane are subtracted from the measured depth values. That way, the spatial depth precision does also contain information about non-linear deformations of the point cloud like e.g. bending. The spatial depth precision for frame *j* is given by

$$depth\_precision\_spatial_j = \mathrm{std}\left(z_i - z_i^{Fit}\right)_j$$

In order to reduce the impact of statistical effects (e.g. dust inside the Validation Box) and to stabilize the result of this validation test point, the above calculation is done for all 20 point clouds and the results are averaged to yield the final result:

$$depth\_precision\_spatial = \frac{1}{20}\sum_{j=1}^{20}\left(depth\_precision\_spatial_j\right)$$

3.2. Temporal Depth Precision (depth_precision_temporal, depth_precision_temporal_q90)

The depth noise *dz* of a single pixel *i* is given by the standard deviation of the measured depth values of 20 point clouds

$$dz_i = \mathrm{std}(z)_i$$

The temporal depth precision (test point "depth_precision_temporal) is then calculated as the mean depth noise of all pixels

$$depth\_precision\_temporal = \mathrm{mean}(dz_i)$$

The quantile 90 of the temporal depth precision (test point "depth_precision_temporal_q90") is calculated as the 90 percentile of the depth noise of all pixels

$$depth\_precision\_temporal\_q90 = \mathrm{quantile}(dz_i)_{90}$$

3.3. Single Shot Depth Error (single_shot_depth_error)

The single shot depth error (test point "single_shot_depth_error") can be understood as a measure for the reliability of the depth measurement. For a single point cloud *j* it evaluates the

mean deviation of all pixels from the ground truth (i.e. the distance from the camera to the planar target) and thus it includes noise as well as systematic effects

$$single\_shot\_depth\_error_j = \mathrm{mean}(z_i - ground\_truth)_j$$

In order to reduce the impact of statistical effects (e.g. dust inside the Validation Box) and to stabilize the result of this validation test point, the above calculation is done for all 20 point clouds and the results are averaged to yield the final result:

$$single\_shot\_depth\_error = \frac{1}{20}\sum_{j=1}^{20}\left(single\_shot\_depth\_error_j\right)$$

3.4. Depth Accuracy (depth_accuracy, depth_accuracy_percent, depth_accuracy_q90_percent)

The depth accuracy is closely related to the single shot depth error but only systematical effects are considered. In order to get rid of all components originating from noise or other statistical effects, the average depth of the 20 point clouds is calculated for each pixel *i*:

$$\bar{z}_i = \mathrm{mean}(z)_i$$

The depth accuracy (test point "depth_accuracy") is then given as the mean difference of all pixels with respect to the ground truth

$$depth\_accuracy\ = \mathrm{mean}(\bar{z}_i - ground\_truth)$$

From this, the relative depth accuracy (test point "depth_accuracy_percent") can be calculated

$$depth\_accuracy\_percent = \frac{depth\_accuracy}{ground\_truth}$$

The quantile 90 of the depth accuracy in percent (test point "depth_accuracy_q90_percent") is calculated by

$$depth\_accuracy\_q90\_percent = \mathrm{quantile}\left(\frac{\bar{z}_i - ground\_truth}{ground\_truth}\right)_{90}$$

3.5. Amplitude Standard Deviation (amplitude_std_temporal, amplitude_std)

For a given distance to the target and given target reflectivity, the amplitude standard deviation, commonly referred to as "amplitude noise", is a measure for the temporal stability of the ToF system. In a first step, the standard deviation of the amplitudes of each single pixel *i* over all 20 amplitude images is calculated

$$da_i = \mathrm{std}(a)_i$$

The amplitude standard deviation (test point "amplitude_std_temporal") is then given as the mean amplitude noise over all pixels

$$amplitude\_std\_temporal = \mathrm{mean}(da_i)$$

The second interpretation of the amplitude standard deviation (test point "amplitude_std") contains information about the uniformity of the measured amplitude over the imager. In a first step, the standard deviation of amplitude values over all pixels is calculated for each frame *j*.

$$da_j = \mathrm{std}(a)_j$$

Then, the mean over all 20 frames is calculated

$$amplitude\_std = \mathrm{mean}(da_j)$$

3.6. Amplitude Mean (amplitude_mean)

The mean amplitude represents the modulated signal strength of the ToF system. Therefore, it is an indicator for ToF depth (noise) performance. In a first step, the temporal average of the amplitudes of each single pixel *i* over all 20 amplitude images is calculated

$$\overline{a_i} = \mathrm{mean}(a)_i$$

The amplitude mean (test point “amplitude_mean”) is then given as the mean amplitude signal over all pixels

$$amplitude\_mean = \mathrm{mean}(\overline{a_i})$$

3.7. Amplitude Min/Max (amplitude_min, amplitude_max)

The test points “amplitude_min” and “amplitude_max” correspond to the mean amplitude of the worst and best performing pixel, respectively. They are calculated by

$$amplitude\_min = \min(\overline{a_i})$$

$$amplitude\_max = \max(\overline{a_i})$$

3.8. Minimum / maximum Amplitude (amplitude_min_of_mins, amplitude_max_of_maxs)

The test points “amplitude_min_of_mins” and “amplitude_max_of_maxs” corresponds to the darkest and brightest pixel within all frames over all pixels, respectively.

$$amplitude\_min\_of\_mins = \min(\min(a)_i)$$

$$amplitude\_max\_of\_maxs = \max(\max(a)_i)$$

3.9. Tilt/Pan of Plane Fit (plane_tilt_angle, plane_pan_angle)

The coordinate values of each single pixel of the point cloud (X, Y, and Z) are based on the intrinsic lens calibration. In order to ensure a correct calibration and to exclude possible hardware defects, not only depth accuracy but also the orientation of the measured target plane is of high interest. In theory, a measured point cloud can be subject to rotations in all three main directions. However, since the Validation Box only consists of a planar, featureless target, rotations around the Z-axis (“Spin”) cannot be detected during the validation process.

In order to determine the orientation of the measured target plane, a planar fit to the data is performed. Here, the temporally averaged point cloud is used. By doing so, the impact of noise or other statistical effect can be reduced. The result of the fitting process is a plane vector, from which possible rotations around the X-axis (“Pan”) and Y-axis (“Tilt”) can be deduced.

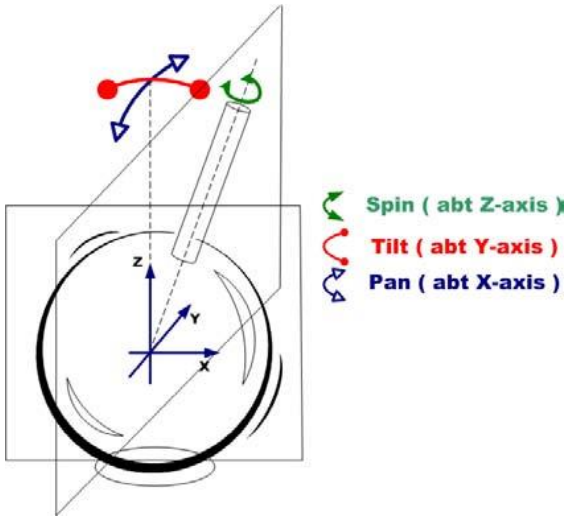

Figure 1: Possible rotations of the target plane

3.10.Number of Calibrated Pixels (number_of_calibrated_pixels_in_roi)

During the ToF calibration, a possible mismatch between the field off illumination (FoI) and the field of view (FoV) of the ToF system is determined and an illumination mask is defined, masking pixels with insufficient lighting. This static illumination mask limits the number of pixels that can be used for the measurement. During validation process, the illumination mask is read from the calibration data and the number of calibrated (non-masked) pixels is evaluated (test point “number_of_calibrated_pixels_in_roi”).

3.11.Confidence (confidence_0 – confidence_7)

In the current implementation of the processing chain for depth calculation, a confidence value regarding the measured depth quality can be defined for each pixel. This value is defined as 0 for low confidence, 1 for high confidence and the intermediate steps from 2 to 7 with increasing confidence. A confidence map exists for each of the recorded frames *j* and the minimum confidence for a single pixel *i* is calculated

$$confidence_i = \min\left(confidences_j\right)_i$$

The number of pixels for each confidence level is then calculated as

$$conficence\_X = \mathrm{sum}\left(confidence_i, \mathrm{where}\ (confidence_i = X)\right)$$

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] pmdtechnologies ag, "AN-SE-3-6-validation_box," 2017.

[2] pmdtechnologies ag, "AN-CAL-3-1-led_box_concept," 2017.

Document History

Document: ToF Validation Test Points [an-se-3-7]

Revision	Origin of Change	Submission Date	Description of Change
0	UFr	2018-08-30	New Application Note
1	MRe	2018-12-07	Update of test points