Detailed Modeling and Calibration of a Time-of-Flight Camera

Christoph Hertzberg, Udo Frese

FB3 – Mathematics and Computer Science, University of Bremen, 28359 Bremen, Germany
{chtz,ufrese}@informatik.uni-bremen.de

Keywords:
Time-of-Flight Camera, Depth Sensor, Sensor Model, Calibration, Robustified Corner Extraction, High Dynamic Range, Lens Scattering

Abstract:
In this paper we propose a physically motivated sensor model of Time-of-Flight cameras. We provide methods to calibrate our proposed model and compensate all modeled effects. This enables us to reliably detect and filter out inconsistent measurements and to record high dynamic range (HDR) images. We believe that HDR images have a significant benefit especially for mapping narrow-spaced environments as in urban search and rescue. We provide methods to invert our model in real-time and gain significantly higher precision than using the vendor-provided sensor driver. In contrast to previously published purely phenomenological calibration methods our model is physically motivated and thus better captures the structure of the different effects involved.

1 INTRODUCTION

Time-of-Flight (ToF) cameras are compact sized sensors which provide dense 3D images at high frame rates. In principle, this makes them advantageous for robot navigation (Weingarten et al., 2004), iterative closest point (ICP)-based 3D simultaneous localization and mapping (SLAM) (May et al., 2009) and as a supporting sensor for monocular SLAM (Castaneda et al., 2011). The advantages come at the cost of many systematic errors, which, to our knowledge, have not been systematically analyzed and modeled so far. Especially for ICP-based SLAM it is crucial to remove or reduce systematic errors.

With the recent introduction of low-cost structured light cameras (most famously the Microsoft Kinect) and the KinectFusion algorithm (Newcombe et al., 2011) most research has shifted away from ToF cameras. Despite the much more complex systematic errors of ToF cameras, we still think they have the important advantages of a more compact size and the ability to detect finer structures. This is because structured light sensors cannot detect objects smaller than the sampling density of the structured light pattern, while ToF cameras can detect even objects of sub-pixel dimension if their reflectance is high compared to the background. Both are important, e.g., in narrow-spaced SLAM scenarios, where we believe ToF cameras have the potential to supersede structured light and stereo vision. In this paper, we focus on the PMD CamBoard nano (PMDtechnologies, 2012), but our results should be transferable to similar sensors.

In the remainder of the introduction, we recapitulate an idealistic sensor model that is often found in the literature and list phenomena where the actual sensor differs from that model. Afterwards, we discuss related work in Sec. 2, followed by Sections 3 to 6 describing the phenomena and their calibration. We verify our results and show the impact of the compensation by means of a practical example in Sec. 7 before finishing with an outlook in Sec. 8.

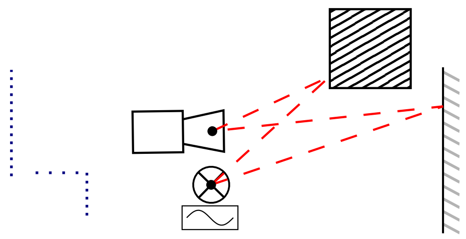


Figure 1: Idealistic Sensor Model. The LED emits modulated light which is reflected by the scene on the right (dashed red paths). From the phase-offset of the received light the sensor can calculate distances and these can be used to build a 3D model of the scene (blue dots on the left). The wave length of the modulation is ideally greater than the longest path lengths expected in the application.

1.1 Idealistic Sensor Model

The general measurement principle of ToF cameras is to send out a modulated light signal and measure its amplitude and phase offset as it returns to the camera (Lange, 2000, §1.3, §2.2), cf. Fig. 1. In the simplest case, the generated signal is a sinusoidal wave with known amplitude a, frequency[1] ν and offset c_O:

$$\psi(t) = a \sin(2\pi\nu t) + c_O \tag{1}$$

As the signal is reflected back into the camera, only a certain ratio α is returned, its phase is shifted proportionally to the time it travelled Δt, and additional constant background light c_B is added to the signal:

$$z(t) = \alpha \cdot \psi(t - \Delta t) + c_B \tag{2}$$

To measure the amplitude and phase of the returned signal, the ToF sensor integrates it over half the phase starting at the k^{th} quarter period, resulting in four images:

$$\begin{aligned} s^{[k]} &= \int_{\frac{k}{4\nu}}^{\frac{k+2}{4\nu}} z(t)\,\mathrm{d}t = \Big[c_1 t - \tfrac{A}{4}\cos\big(2\pi\nu(t-\Delta t)\big)\Big]_{\frac{k}{4\nu}}^{\frac{k+2}{4\nu}} \\ &= c_2 + \tfrac{A}{2}\cos\big(\tfrac{\pi}{2}k - 2\pi\nu\Delta t\big), \end{aligned} \tag{3}$$

with $A = \frac{2\alpha a}{\pi\nu}$. Actually, the signal is integrated over multiple half-phases for a much longer exposure time t_I. Mathematically, this corresponds to multiplying the emitted and received amplitude of the signal (as well as all constants $c_\bullet$) by t_I.

The values of the constants $c_\bullet$ are not important, because they will cancel out in the following equations. From (3) the amplitude A and phase offset Δt of the received signal can be calculated using the well-known formulae:

$$A = \sqrt{\big(s^{[0]} - s^{[2]}\big)^2 + \big(s^{[1]} - s^{[3]}\big)^2} \tag{4a}$$

$$\Delta t \equiv \tfrac{1}{2\pi\nu}\operatorname{atan2}(s^{[1]} - s^{[3]}, s^{[0]} - s^{[2]}) \mod \tfrac{1}{\nu}. \tag{4b}$$

The phase offset can only be determined modulo the length of the phase. Knowing Δt it is possible to calculate the travelled distance by multiplying with the speed of light. Note that we can interpret $\big(s^{[0]} - s^{[2]}, s^{[1]} - s^{[3]}\big)$ as a complex number with magnitude A and phase $2\pi\nu \cdot \Delta t$.

1.2 Suppression of Background Illumination

When measuring the subimages directly as in (3), the constant offset c_2 can become much higher than the usable amplitude A, especially if the amount of c_B is high. This reduces the signal-to-noise ratio (SNR) of the differences $s^{[k]} - s^{[k+2]}$ and can even lead to saturation in the $s^{[k]}$.

To reduce this effect, the sensor integrates $s^{[k]}$ and $s^{[k+2]}$ into two separate bins A and B alternately and instantaneously compensates saturation effects by simultaneously adding equal charges to both bins (Möller et al., 2005, §4.2). After integration, the difference between both bins is determined inside the chip and read out.

This technique is called Suppression of Background Illumination (SBI) by the manufacturer. While it significantly improves the SNR it also makes modeling the sensor more complicated.

1.3 Diversions from the Idealistic Model

We determined several phenomena which deviate from the idealistic model. Here we list all phenomena of which we are aware and reference to future sections where they will be handled in detail or we justify how and why we can neglect them in the scope of this paper. In describing the model we always follow the path of the light from *emission* over scene interaction and *optics* to its *detection* in the sensor.

Emission will describe all effects from wave generation until the light interacts with the scene.

- **Light Modulation** of the LED is not a sinusoidal wave as in (1), but closer to a rectangular wave. We model this in Sec. 4.1.
- **Temperature** of the LED has an influence on the shape of the modulated light. In this paper we will neglect this effect and work around it by keeping the temperature of the LED at a constant level of 50 °C, by adjusting its activity.
- **Vignetting** happens due to the non-uniform light distribution of the lens in front of the LED. We model this in Sec. 4.2.

Optical Effects happen before the light hits the sensor array.

- **LED Position** relative to the camera is often ignored. Especially, for narrow-spaced environments its position has a measurable effect. We model this and measure the position manually.
- **Multi-Path Reflections** can happen when light is reflected multiple times inside the scene. Especially, in sharp corners this can

[1] Our camera has a fixed frequency of $\nu = 30\,\text{MHz}$ which results in a wavelength of $\lambda = c/\nu \approx 10\,\text{m}$.

have a significant impact. In this paper, we will only discuss it briefly in Sec. 8.3.

- **Flying Pixels** occur at depth discontinuities in the scene leading to "ghost pixels" and are mentioned briefly in related work.
- **Lens Distortion** is a well-understood problem of cameras and will be handled in Sec. 3.
- **Lens Scattering** happens when light entering the lens is reflected inside the lens casing and captured by neighboring pixels. We discuss and handle this phenomenon in Sec. 6.

Detection models everything happening after the light passed through the optics.

- **Non-Linearities** occur mostly during integration of the received light. We model this by a non-linear photon response curve described in Sec. 4.4.
- **Fixed Pattern Noise** (FPN) occurs due to chip-internal different signal propagation times as well as slightly different gains for each pixel. This is modeled in Sec. 4.5.
- **Background Illumination** can have a significant effect on the recorded signal. It will be ignored in this paper as it is negligible in our case, but we discuss it in Sec. 8.2.

2 RELATED WORK

Monocular camera calibration can be considered a "solved problem" with ready-to-use calibration routines, e.g., in the library (OpenCV, 2014).

All ToF calibration related papers we found, have in common that they calculate an a-priori offset using a formula such as (4b) and compensate the deviation of this value towards a ground-truth value using various methods. This means, unlike us, they only use Δt not A as input to correct the measured distances.

(Kahlmann et al., 2006) used a grid of active NIR LEDs for intrinsic camera calibration. They then used a high accuracy distance measurement track line to obtain ground-truth distances and median-filter these measurements to generate a look-up table depending on integration time and distance to compensate deviations from the ground-truth. They already observed the influence of self-induced temperature changes on the measured distances.

(Lindner and Kolb, 2006) fit the deviation to a ground-truth distance using a global B-spline. This was then refined using a linear per-pixel fit.

Both (Fuchs and May, 2008; Fuchs and Hirzinger, 2008) attached their camera to an industrial robot to get ground-truth distances towards a calibration plane. They fit the deviation to the ground-truth using a set of splines depending on the distance with different splines for different amplitude ranges.

In our opinion, neither of the above methods is capable of properly modeling non-linearities and will suffer decreasing accuracy as soon as different integration times or large variances in reflectivity occur. Furthermore, they are not able to automatically detect saturation effects (although we assume they might detect them a-priori) or calculate HDR images from a set of images with different integration times.

In other related work, (Mure-Dubois and Hügli, 2007) propose a way to compensate lens scattering by modeling a point spread function (PSF) as a sum of Gaussians. In general, we will use a similar approach. However, we are capable of making more precise calibration measurements using HDR images and we use a combination of many more Gaussian kernels, which we calibrate exploiting the linearity of the scattering effect.

The flying pixel effect has often been addressed, e.g., (Sabov and Krüger, 2010) present different approaches to identify and correct flying pixels. Also, (Fuchs and May, 2008) give a simple method to filter out "jump edges".

Finally, multi-path reflections are handled by (Jimenez et al., 2012). Their iterative compensation method is very computationally expensive and goes beyond the scope of this paper.

3 INTRINSIC AND EXTRINSIC MONOCULAR CALIBRATION

In order to calibrate the actual sensor, we first calibrated the camera intrinsics and obtained ground truth positions of the camera relative to a checkerboard target. All operations are solely based on amplitude images obtained from the camera as with an ordinary camera. Amplitudes are calculated as magnitude from real and imaginary part according to the simple model (4a).

As monocular camera calibration is a mature and reliable procedure we use the result a) for the monocular intrinsic part of our model that maps 3D points to pixels b) to obtain distance values for each pixel to calibrate the depth related part of the model and c) as ground-truth for evaluating the results in Sec. 5. We believe that using the same technique for b) and c) is valid since

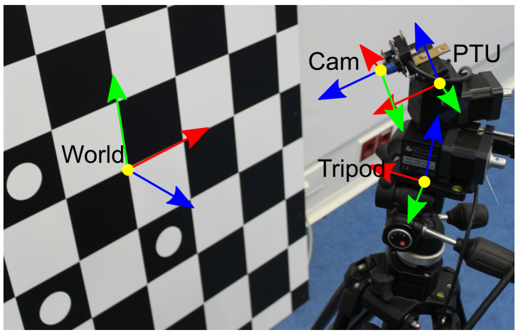


Figure 2: Setup of checkerboard calibration. The `World` coordinate system lies at the center of the checkerboard with x and y axes pointing along the grid and z pointing upwards. The PTU has two coordinate systems which are rotated towards each other by the current angles of the PTU. We define the coordinate system attached to the tripod as `Tripod` with x pointing forward, y pointing left and z pointing upwards along the pan axis. The rotating system is called `PTU` and actually shares the position of `Tripod` but is panned and tilted. Finally, we have the camera coordinate system `Camera` which is centered at the camera's optical centre with z pointing forward along the optical axis and x and y pointing right and down (thus matching pixel coordinates). Coordinate systems are overlayed using colors red, green, and blue for x, y, and z-axis.

the technique is highly trustworthy and applied to different images.

3.1 Experimental Setup

Determining intrinsic camera parameters can be considered a standard problem. It is commonly solved by checkerboard calibration, e.g., using the routines provided by (OpenCV, 2014). Having a wide-angle, low resolution camera requires an increased number of corner measurements to get an adequate precision. To achieve this with reasonable effort, we attached the camera to a pan-tilt-unit (PTU) mounted on a tripod (see Fig. 2). We use a $1\,\mathrm{m} \times 1\,\mathrm{m}$ checkerboard with $0.125\,\mathrm{m}$ grid distance, resulting in 8×8 corner points per image. Using the PTU helps to partially automatize the calibration process, by providing lots of corner measurements without having to manually reposition the checkerboard or camera. Furthermore, it increases the accuracy of the calibration, by having more measurements defining the pose of the tripod (and thereby indirectly each camera pose) relative to the checkerboard.

Before starting the calibration, we manually adjusted the focal length of the camera, viewing a Siemens star, optimizing focus at about $1\,\mathrm{m}$.

3.2 Measurement Equations

From here on, we will use the notation $T_{A \leftarrow B}$ to describe the transformation from coordinate system B to A. We abbreviate the `World`, `Tripod`, `PTU` and `Camera` systems introduced in Fig. 2 by their first letters and append indices where required.

We have a sequence of tripod poses $T_{W \leftarrow T_i}$ and for each tripod pose we have a sequence of PTU rotations $T_{T_i \leftarrow P_{ij}}$. Furthermore, there is a transformation from the camera coordinate system to the PTU $T_{P \leftarrow C}$ (this is assumed to be the same for all measurements). We have a set of corner points with known world coordinates $x_W^{[k]}$ which we transform to camera coordinates via

$$x_{C_{ij}}^{[k]} = (T_{W \leftarrow T_i} \cdot T_{T_i \leftarrow P_{ij}} \cdot T_{P \leftarrow C})^{-1} x_W^{[k]}. \tag{5}$$

To map a point from 3D camera coordinates to pixel coordinates, we use the OpenCV camera model (OpenCV, 2014, "calib3d") described by this formula for $\left[\begin{smallmatrix} u_1 \\ u_2 \end{smallmatrix}\right] = \varphi_K([x_1\ x_2\ x_3]^\top)$:

$$\left[\begin{smallmatrix} w_1 \\ w_2 \end{smallmatrix}\right] = \frac{1}{x_3} \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right] \qquad \text{“Pinhole model”} \tag{6a}$$

$$\left[\begin{smallmatrix} v_1 \\ v_2 \end{smallmatrix}\right] = \kappa \cdot \left[\begin{smallmatrix} w_1 \\ w_2 \end{smallmatrix}\right] + \left[\begin{smallmatrix} 2p_1 w_1 w_2 + p_2(r^2 + 2{w_1}^2) \\ p_1(r^2 + 2{w_2}^2) + 2p_2 w_1 w_2 \end{smallmatrix}\right] \tag{6b}$$

with $r^2 = w_1^2 + w_2^2$, $\kappa = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$

$$\left[\begin{smallmatrix} u_1 \\ u_2 \end{smallmatrix}\right] = \left[\begin{smallmatrix} f_1 v_1 \\ f_2 v_2 \end{smallmatrix}\right] + \left[\begin{smallmatrix} c_1 \\ c_2 \end{smallmatrix}\right] =: \varphi_K(x), \tag{6c}$$

where we join all intrinsic calibration parameters into a vector $K = [f_1, f_2, c_1, c_2, k_1, k_2, p_1, p_2, k_3]$.

The transformations $T_{P \leftarrow C}$ and each $T_{W \leftarrow T_i}$ are modeled as unconstrained 3D transformations, the PTU orientations $T_{T_i \leftarrow P_{ij}}$ as 3D orientations which are initialized by and fitted to orientations $\hat{P}_{ij}$ measured by the PTU. The projected corner positions are fitted to corner positions $\hat{z}_{ij}^{[k]}$ extracted from the amplitude images.

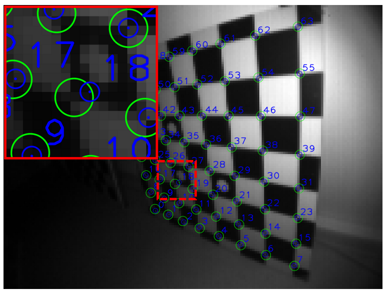
As usual for model fitting, we solve a nonlinear least squares problem, minimizing the following functional:

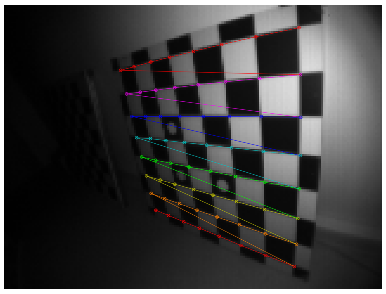
$$F(K, T_{P \leftarrow C}, [T_{W \leftarrow T_i}]_i, [T_{T_i \leftarrow P_{ij}}]_{ij}) = \tag{7}$$

$$\frac{1}{2} \sum_{i,j,k} \left\| \hat{z}_{ij}^{[k]} - \varphi_K(x_{C_{ij}}^{[k]}) \right\|^2 + \frac{1}{2} \sum_{i,j} \left\| T_{T_i \leftarrow P_{ij}} - \hat{P}_{ij} \right\|^2$$

with $\varphi_K(x_{C_{ij}}^{[k]})$ according to (5) and (6). We assume that the PTU is accurate approximately to its specified resolution $\frac{2\pi}{7000} \approx 0.05°$. This corresponds to about $0.08\,\mathrm{px}$ and as this is about the same magnitude as sub-pixel corner detection, we cannot neglect it. Therefore, the second term is scaled accordingly.

F is minimized using our general purpose least squares solver SLoM (Hertzberg et al., 2013).


(a) Failed extraction


(c) Reprojected corners

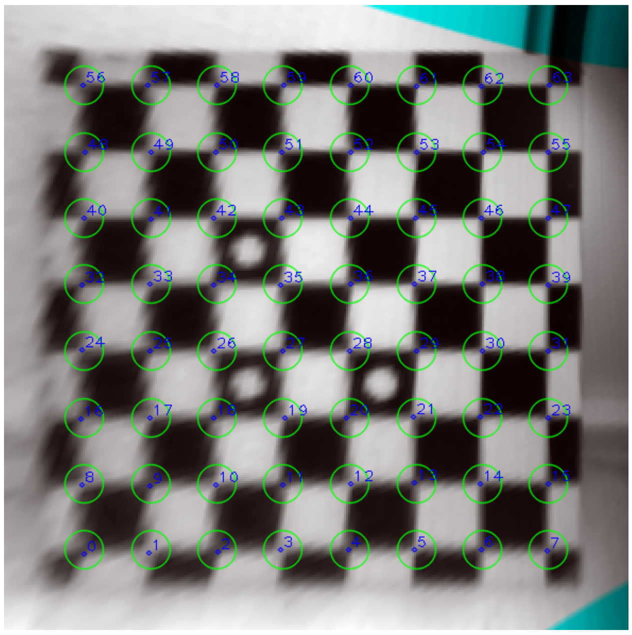

(b) Corners extracted in rectified image

Figure 3: Robustified corner extraction. Green circles show the initial guess and search window for corner refinement, blue circles show the refined corner positions. (a) Failed corner extraction, because of unrelated gradients in the search window. Note that for smaller search windows the extraction of the top-right corners would lose precision. (b) Using an initial calibration and a guess of the camera pose the image is projected to the checker-board plane. We also adjust the brightness based on the expected distance and normal vector (cf. Sec. 4.3) Areas outside the original image are continued using the border pixels (cyan/black areas in this figure). Note that even the bottom-left corners are robustly extracted. (c) The refined corner positions are back-projected to the original image.

3.3 Robustified Corner Extraction

To extract corners we use the `cornerSubPix` method from OpenCV. Applying the method directly to the amplitude image turned out to be unstable if corners are close to each other or to unrelated image gradients (Fig. 3a). The reason is that gradients from neighboring edges are wrongfully included into the refinement. We have considered masking out invalid edges from the image, but that would have required massive rewriting of the corner detection. Instead, we project the image to the plane where the checkerboard should be (adding some border pixels on each side, Fig. 3b), extract the corners in that image and back-project the found corners to the original image (Fig. 3c). An advantage of this procedure is that the corner extractor can always work using the same search window. By taking the back-projected corner positions as measurements we automatically have their accuracies weighted according to the amount of pixels involved in the measurement.

This procedure requires an initial guess of the camera intrinsics, which is taken from a previous calibration, and the pose of the camera relative to the checkerboard, which is determined by calculating the relative pose from a previous PTU state or by manually selecting four points in the center of the checkerboard. To ensure robustness we decided to not rely on automatic checkerboard detection if the camera pose is entirely unknown. Still, the only case that requires user interaction is whenever the pose of the tripod has been changed.

3.4 Calibration Results

We recorded 90 frames from two tripod poses by using 45 different PTU orientations each. For each frame we recorded 8 images with 4096 µs exposure time[2] and calculated the amplitude of the mean (complex) image.

Overall, we extracted more than 4500 corners which means that about 50 of 64 corners were

[2]N.B.: Exposure times above 2000 µs require calling an undocumented driver command

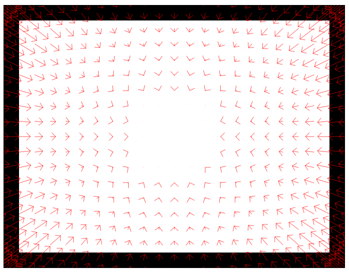


Figure 4: Lens Distortion. The arrows point from the theoretical undistorted point to the distorted point. Distortions below 1 px are not shown. The image border is marked black. As common for wide-angle lenses the distortion near the borders is very high.

visible on average. Optimization lead to a root-mean-squares (RMS) error of about 0.13 px. The estimated focal length was $f_1 \approx f_2 \approx 89.5\,\text{px}$ with the camera center at $c \approx \left[\begin{smallmatrix}79.5\,\text{px}\\60.5\,\text{px}\end{smallmatrix}\right]$. The resulting distortion pattern is shown in Fig. 4.

The calibration was repeated with a different set of poses and resulted in the same parameters with less than 0.5 % discrepancy. We considered this accurate enough for our purpose.

4 PROPOSED PIXEL MODEL

In this section we propose a physically motivated sensor model and fit it as well as possible to the data recorded in the previous section. It is based on (Lange, 2000; Möller et al., 2005) and some educated guesses on the underlying mechanisms.

As in Sec. 1.3 we follow in the description the path of light from the LED through the scene to the sensor. We assume that the measurement of each single pixel is solely caused by the reflection of a surface patch with normal n and a position x^{Cam} relative to the camera center. We also assume that the surface patch is a perfect Lambertian reflector with albedo α. Knowing the position of the LED relative to the camera d^{LED}, we determine the position of the surface patch relative to the LED $x^{\text{LED}} := x^{\text{Cam}} - d^{\text{LED}}$. This is visualized in Fig. 5.

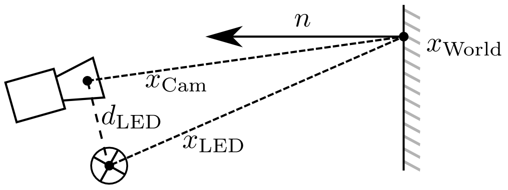


Figure 5: Path of the light from the LED, reflected by a white wall to the camera. The position of the LED relative to the camera is d_{LED}, and $x_\bullet$ denotes the coordinates of the wall patch in each coordinate system. The normal vector of the surface is called n.

4.1 Emitted Light Wave

First of all, the emitted light wave is not a sinusoidal. By design, it is closer to a rectangular wave, but due to hardware restrictions it is smoothed near the switchover points and not necessarily constant elsewhere. All we assume is that the emitted light can be described by a continuous function $\psi(t)$ with a periodicity $\frac{1}{\nu}$.

For simplicity we can assume that $\nu = \frac{1}{4}$ (by arbitrarily choosing the unit of ν) and therefore $\psi(t) = \psi(t+4)$. We will also not model ψ directly but its integral Ψ over half the phase. By subtracting half the integral over the entire phase, Ψ becomes anti-periodic with $\Psi(t) = -\Psi(t+2)$:

$$\Psi(t) := \int_t^{t+2} \psi(\tau)\,\mathrm{d}\tau - \underbrace{\frac{1}{2}\int_0^4 \psi(\tau)\,\mathrm{d}\tau}_{=:\Psi_0}. \tag{8}$$

Because the scale and time offset of ψ and thus Ψ are arbitrary, we choose them so that $\Psi(0) = 0$ and $\Psi(1) = 1$. We will then model Ψ using two polynomials Ψ_s and Ψ_c of order 8 with:

$$\Psi(t) = \begin{cases} \Psi_\text{s}(t) & |t| \le \frac{1}{2} \\ \Psi_\text{c}(t-1) & |t-1| \le \frac{1}{2} \\ -\Psi(t-2) & t > \frac{3}{2} \\ -\Psi(t+2) & t < -\frac{1}{2}, \end{cases} \tag{9}$$

where the transitions between Ψ_s and Ψ_c are modeled to be twice continuously differentiable. The resulting Ψ is later shown in Fig. 7.

4.2 Vignetting

The lens in front of the LED does not distribute the light uniformly which causes strong vignetting. We model this effect depending on the direction of the light exiting the LED by a bi-polynomial ℓ depending on $\frac{1}{x_3^{\text{LED}}}\left[\begin{smallmatrix}x_1^{\text{LED}}\\x_2^{\text{LED}}\end{smallmatrix}\right]$, with

$$\ell(\left[\begin{smallmatrix}u_1\\u_2\end{smallmatrix}\right]) = \sum_{i=0}^{3}\sum_{j=0}^{i} l_{j,i-j}\,u_1^j u_2^{i-j}. \tag{10}$$

We write $\ell(x) := \ell(\frac{1}{x_3}\left[\begin{smallmatrix}x_1\\x_2\end{smallmatrix}\right])$ and collect the coefficients into an upper-left matrix $L = (l_{ij})_{i+j\le 3}$, where we normalize $l_{00} := 1$.

4.3 Scene Interaction

If we assume perfect Lambertian reflection, the amount of reflected light is proportional to the albedo $\alpha_{i,p}$ and the scalar product of the direction of incoming light $x_{i,p}^{\mathrm{LED}} / \left\| x_{i,p}^{\mathrm{LED}} \right\|$ with the surface normal n_i (whenever a value depends on a camera pose T_i or a pixel p, we will from now denote this by indexes $_{i,p}$). Also, it is reciprocally proportional to the squared distance from the LED.

We combine this with the vignetting from the previous section and the integration time t_I to calculate the expected effective amplitude

$$A_{i,p}^{[t_I]} = t_I \cdot \alpha_{i,p} \cdot \ell(x_{i,p}^{\mathrm{LED}}) \frac{\langle x_{i,p}^{\mathrm{LED}}, n_i \rangle}{\left\| x_{i,p}^{\mathrm{LED}} \right\|^3}. \qquad (11)$$

Note that $\alpha_{i,p}$ will contain the normalization factors of the emitted light ($\Psi(1) = 1$) and the vignetting ($\ell(0) = 1$).

The phase offset is calculated by dividing the distance travelled $\left\| x_{i,p}^{\mathrm{LED}} \right\| + \left\| x_{i,p}^{\mathrm{Cam}} \right\|$ by the speed of light. As the speed of light equals the product of wavelength λ and frequency ν, and we chose $\nu = \frac{1}{4}$ (cf. Sec. 4.1) this can be simplified to

$$\Delta t_{i,p} = 4 \cdot \frac{\left\| x_{i,p}^{\mathrm{LED}} \right\| + \left\| x_{i,p}^{\mathrm{Cam}} \right\|}{\lambda}. \qquad (12)$$

4.4 Photon Response Curve

The actual pixel measurements are not linear in the received light. As in (Lange, 2000) we assume that each pixel consists of two "bins" A and B. We assume that each bin has a response-curve g^{A}, g^{B} and the pixel measurement is the difference between those bins offset by γ. Bin A is assumed to accumulate for $0 \le t < 2$ and B for $2 \le t < 4$

$$\begin{aligned} s_{i,p}^{[k]} &= \gamma_p + g_p^{\mathrm{A}} \left(\int_k^{k+2} z_{i,p}(t) \right) \mathrm{d}t - g_p^{\mathrm{B}} \left(\int_{k+2}^{k+4} z_{i,p}(t) \right) \mathrm{d}t \\ &= \gamma_p + g_p^{\mathrm{A}} \left(A_{i,p}^{[t_I]} (\Psi_0 + \Psi(\Delta t_{i,p} + k)) \right) \\ &\quad - g_p^{\mathrm{B}} \left(A_{i,p}^{[t_I]} (\Psi_0 - \Psi(\Delta t_{i,p} + k)) \right). \end{aligned} \qquad (13)$$

By design, g^{A} and g^{B} are almost identical and as their values cannot be measured individually we make a simplification by subtracting $s_{i,p}^{[k]} - s_{i,p}^{[k+2]}$. This cancels out the γ_p and g_p^{A} and g_p^{B} can be joined to a single function $g_p^{\mathrm{S}}(x) := g_p^{\mathrm{A}}(x) + g_p^{\mathrm{B}}(x)$:

$$\begin{aligned} s_{i,p}^{[k]} - s_{i,p}^{[k+2]} = \; & g_p^{\mathrm{S}} \left(A_{i,p}^{[t_I]} (\Psi_0 + \Psi(\Delta t_{i,p} + k)) \right) \\ & - g_p^{\mathrm{S}} \left(A_{i,p}^{[t_I]} (\Psi_0 - \Psi(\Delta t_{i,p} + k)) \right). \end{aligned} \qquad (14)$$

We model g_p^{S} using a rational polynomial

$$g_p^{\mathrm{S}}(x) = x \frac{x + g_0}{x + g_1}, \qquad (15)$$

where we assume that the coefficients between pixels are similar enough to be modeled by the same coefficients $G = [g_0, g_1]$. Finally, we capture (14) into a helper function:

$$\begin{aligned} g^{\mathrm{D}}(A, \Psi_0, \Psi) &= g^{\mathrm{S}}(A(\Psi_0 + \Psi)) - g^{\mathrm{S}}(A(\Psi_0 - \Psi)) \\ &= 2A\Psi \cdot \left(1 + \tfrac{g_1(g_0 - g_1)}{A^2\Psi^2 - (A\Psi_0 + g_1)^2} \right). \end{aligned} \qquad (16)$$

4.5 Fixed Pattern Noise (FPN)

Due to different internal signal propagation times, individual pixels have a time-offset δt_p, which must be added to the time offsets $\Delta t_{i,p}$ before being inserted into (14). Furthermore, pixels have a slightly different gain, which we model by multiplying each g_p by a factor $h_p \approx 1$.

4.6 Complete Pixel Model

Summarizing the previous subsections, we model each complex-valued pixel at p, with

$$I_{i,p}^{[t_I]} = \left(s_{i,p}^{[0]} - s_{i,p}^{[2]}, s_{i,p}^{[1]} - s_{i,p}^{[3]} \right) \qquad (17)$$

$$\begin{aligned} &= h_p \cdot \left(g^{\mathrm{D}}(A_{i,p}^{[t_I]}, \Psi_0, \Psi(\Delta t_{i,p} + \delta t_p)), \right. \\ &\qquad \left. g^{\mathrm{D}}(A_{i,p}^{[t_I]}, \Psi_0, \Psi(\Delta t_{i,p} + \delta t_p + 1)) \right). \end{aligned} \qquad (18)$$

Overall, the sensor model depends on the coefficients of the emitted light wave $\Psi_{\mathrm{c}}^{[k]}$, $\Psi_{\mathrm{s}}^{[k]}$ and Ψ_0 which we collect into a vector Ψ_{p}, the parameters L of the vignetting polynomial, the coefficients G of the photon response curve, and the FPN images $H = (h_p)_p$ and $\delta T = (\delta t_p)_p$.

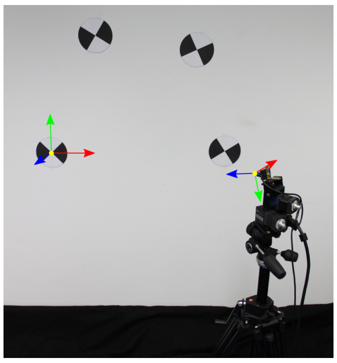
5 PIXEL CALIBRATION

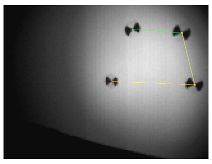
In this section we describe how we calibrate the previously proposed model. We also show how our model can be used to obtain high dynamic range (HDR) images and propose a way to invert our model, i.e., to actually compute distance values using the measurements of the sensor.

5.1 White Wall Recordings

For calibration of each sensor pixel we need many measurements of equal albedo $\alpha_{i,p} = \alpha$ and with a known position and surface normal relative to the camera and the LED. We chose to record an ordinary white-painted wall, to which we attached markers as illustrated in Fig. 6a.

We measured the distances between the markers using a tape measure. We defined the wall


(a) Setup, Coordinate systems


(b) Corners

(c) White mask

Figure 6: White wall with markers. We covered the floor with low-reflective textile to limit multi-path reflections.

coordinate system to originate from the first marker, with the second marker in x-direction and both others on the x-y-plane. Having four markers and six distances between them, we have an overdetermined system. Also, viewing the four markers in a camera image is sufficient to determine the pose of the camera relative to the wall.

As in Sec. 3, we recorded the wall using the same tripod position for various PTU orientations. This again increases the number of constraints defining the tripod pose relative to the plane and also semi-automatizes the process of obtaining calibration values. For each PTU orientations we made recordings with various integration times, starting from 16 µs up to 4096 µs.

The camera poses where optimized using the same functional (7) as in Sec. 3, but this time fixing the camera intrinsics K and the transformation from camera to PTU coordinates $T_{P \leftarrow C}$. The resulting camera poses will be considered as groundtruth from now on and we will simply call them T_i, ignoring the state and index of the PTU.

From the already known camera intrinsics K we can calculate a direction vector v_p for each pixel p. Using the camera pose T_i we can calculate the vector $x_{i,p}^{\text{Cam}}$ going from the camera center to the center of the corresponding patch at the wall.

Knowing also the displacement d^{LED} of the LED relative to the camera[3], allows us to calculate $x_{i,p}^{\text{LED}} = x_{i,p}^{\text{Cam}} - d^{\text{LED}}$ (cf. Fig. 5).

The camera pose and intrinsics are also used to project the positions of the markers into the image in order to exclude these pixels from the calibration by providing a mask M_i of pixels assumed to be white (cf. Fig. 6c).

[3]This is only measured manually at the moment but small inaccuracies have negligible influence here.

5.2 Optimization

For now, we assume that all pixel measurements are independent, i.e., we neglect influence of lens scattering and multi-path reflections.

We use the ground truth values from the previous subsection and the measured images for different poses T_i and different integration times t_I, viewed as complex images

$$\hat{I}_{i,t_I} = (\hat{s}_{i,t_I}^{[0]} - \hat{s}_{i,t_I}^{[2]}, \hat{s}_{i,t_I}^{[1]} - \hat{s}_{i,t_I}^{[3]}), \tag{19}$$

to assemble a functional

$$F(\Psi_{\text{p}}, G, L, H, \delta T) = \sum_{i,t_I,p} \left| I(p) - \hat{I}_{i,t_I}(p) \right|^2 \tag{20}$$

We alternately optimize Ψ_{p}, G and L using (Hertzberg et al., 2013) and the FPN parameters H, δT using a linearized least squares procedure per pixel.

5.3 Inverse Model

To invert the model we need to solve (18) for the unknown amplitude $A_{i,p}^{[t_I]}$ and phase-shift $\Delta t_{i,p}$. Since these calculations are independent for all pixels and poses, we abbreviate $A = A_{i,p}^{[t_I]}$ and

$$d_0 = \hat{s}_{i,t_I}^{[0]} - \hat{s}_{i,t_I}^{[2]} \qquad d_1 = \hat{s}_{i,t_I}^{[1]} - \hat{s}_{i,t_I}^{[3]}, \tag{21}$$

$$\Phi_0 = A\Psi(\Delta t_{i,p}) \qquad \Phi_1 = A\Psi(\Delta t_{i,p} + 1). \tag{22}$$

At first we ignore the fixed-pattern offset $\delta t_{i,p}$ and only solve for Φ_0, and Φ_1, using the fact that $|\Phi_0| + |\Phi_1| \approx A$. This is done using a fixpoint iteration based on (16) starting with

$$\Phi_0 = d_0 \qquad\qquad \Phi_1 = d_1. \tag{23}$$

The next step is to compute Δt_p from Φ_0 and Φ_1. For that we need the pseudo-atan corresponding to the pseudo-sine Ψ_{s} and pseudo-cosine Ψ_{c}. This is obtained during the calibration by once fitting an 8th order polynomial Ψ_{a} to $\Psi_{\text{a}}(\Psi_{\text{s}}(t)/\Psi_{\text{c}}(t)) \approx t$ and then use it here in the inverse model.

5.4 High-Dynamic-Range

Capturing multiple recordings with different integration times of a static scene we can use our model to produce high dynamic range (HDR) images. This is done using a similar fixpoint iteration as in the previous subsection. Indeed the previous method can be seen as a special case, where only a single image is provided.

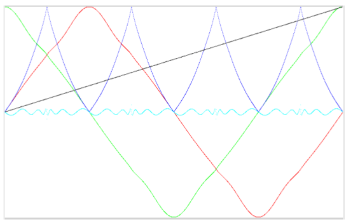


Figure 7: Integrated light wave Ψ modeled as pseudo sine Ψ_s (red) and pseudo cosine Ψ_c (green). Ψ is anti-periodic with $\Psi(t) = -\Psi(t+2)$. Ψ_s and Ψ_c are modeled as polynomials with $\mathcal{C}^2$-transitions at $k + \frac{1}{2}$, for $k \in \mathbb{Z}$. The blue graph is the ratio of sine over cosine, and the black graph is the pseudo-atan calculated from the ratio and shifted according to the signs of Ψ. The turquoise line is the deviation of the pseudo-atan from the actual inverse, scaled by 100.

5.5 Results

In Fig. 7 we show the graph of the calibrated pseudo-sine, -cosine and -atan function. We see that the function is closer to a triangular wave than to a sine wave, indicating that the original signal is close to a rectangular wave.

6 LENS SCATTERING

The lens that comes with the Camboard Nano can be considered as rather low-cost and has significant lens scattering and lens flare effects. What happens is that light entering the lens is reflected inside the lens casing and captured by neighboring pixels or entirely different pixel. For ToF cameras this not only results in slight amplitude changes, but more importantly in distortions of the measured phases (and thus distances) towards brighter light sources, because pixel phases are a mixture of the actual phase and a halo created by a strong light source with a phase corresponding to the distance of the light source.

6.1 Measuring Scattering Effects

To measure the lens scattering, we set up an area of highly non-reflective textile and placed a retro-reflecting disc inside it. Then we made multiple high-dynamic-range recordings of this area once with the retro-reflecting disc and once without it. We calculated the complex valued difference image – which under ideal circumstances would only show the disc. Fig. 8a shows an example.

In these difference images D_i we declared every pixel $u \in \Omega$ above a threshold Θ as being part of the blob B_i causing the scattering. Images with too few blob pixels are ignored.

$$B_i(u) = \begin{cases} D_i(u) & D_i(u) > \Theta \\ 0 & \text{else.} \end{cases} \tag{24}$$

Everything outside this blob and a safety radius of 2 px is assumed to be caused by lens scattering. For that we define a mask set M_i:

$$M_i = \left\{ u \in \Omega;\ \forall_{v, \|v-u\|<2} : D_i(v) \leq \Theta \right\}. \tag{25}$$

6.2 Scattering Model

In this paper we restrict to point symmetrical and translation invariant scattering, leaving out non-symmetries and lens flare effects.

This can be modeled as convolution with a point spread function (PSF) Q for which

$$D_i = B_i * Q, \tag{26}$$

where $*$ expresses a convolution operation with zero border condition:

$$(B * Q)(u) = \int_{v\in\Omega} B(v) \cdot Q(u-v)\, \mathrm{d}v. \tag{27}$$

For brevity, we define a (semi) scalar product and corresponding semi-norm[4] over M_i as

$$\langle f, g \rangle_{M_i} = \sum_{u \in M_i} \overline{f(u)} g(u), \tag{28}$$

$$\|f\|_{M_i} = \sqrt{\langle f, f \rangle_{M_i}}. \tag{29}$$

We then try to find a PSF Q which minimizes

$$F(Q) = \tfrac{1}{2} \sum_j \|D_i - (B_i * Q)\|^2_{M_i}, \tag{30}$$

where we model Q as a sum of Gaussian-shaped functions Q_j with a fixed set of σ_j

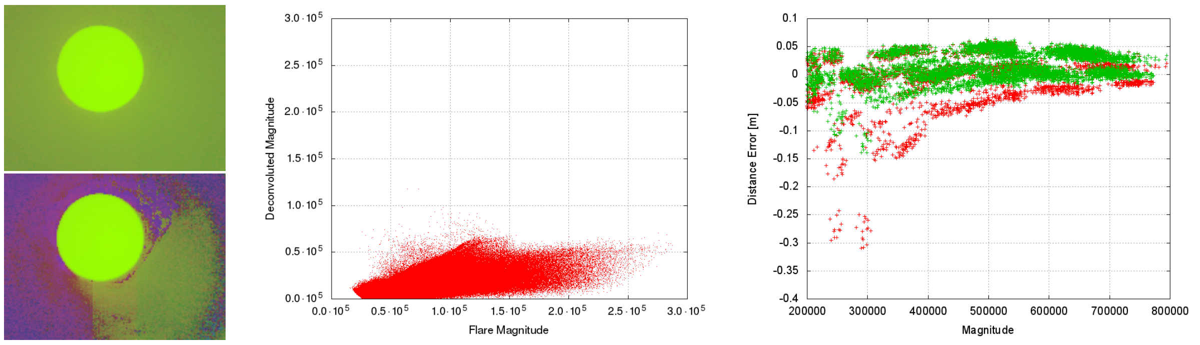
$$Q(u) = \sum_j q_j Q_j(u) = \sum_j q_j \exp\left(-\tfrac{|u|^2}{2\sigma_j^2}\right). \tag{31}$$

Inserting (31) into (30) and exploiting the linearity of convolutions and the bilinearity of the scalar product, we calculate

$$F(Q) = \tfrac{1}{2} \sum_i \Big(\|D_i\|^2_{M_i} - 2 \langle D_i, B_i * Q \rangle_{M_i} + \|B_i * Q\|^2_{M_i} \Big) \tag{32}$$

$$= \tfrac{1}{2} \underbrace{\textstyle\sum_i \|D_i\|^2_{M_i}}_{=:c} - \sum_j \underbrace{\textstyle\sum_i \langle B_i * Q_j, D_i \rangle_{M_i}}_{=:b_j} q_j + \tfrac{1}{2} \sum_{j,k} \underbrace{\textstyle\sum_i \langle B_i * Q_j, B_i * Q_k \rangle_{M_i}}_{=:A_{jk}} q_j q_k. \tag{33}$$

[4]Because we are masking out pixels, we can have non-zero images with zero norm.



(a) Original and Deconvoluted test image (b) Magnitude of error in test images before and after deconvolution (c) Distance difference to ground-truth before (red) and after (green) deconvolution

Figure 8: Figures (a) and (b) show the results of deconvoluting the training images. Figure (c) shows the difference to a checkerboard ground-truth distance, before and after compensating the scattering effect dependent on the magnitude of received light.

Collecting the coefficients q_j into a vector q this leads to a simple linear least squares problem:

$$F(Q) = \min_{Q}! \Leftrightarrow \tfrac{1}{2}q^\top Aq - b^\top q + \tfrac{1}{2}c = \min_{q}! \quad (34)$$

$$\Leftrightarrow Aq = b. \quad (35)$$

6.3 Inverse Lens Scattering

The previous subsections described how we estimated the point spread function, describing the scattering. We will now show how this is compensated using Fourier transformations. The measured image $\tilde{I}$ can be computed from the unscattered image as $\tilde{I} = I + (I * Q)$. Using the identity $\delta * I = I$ and applying the convolution theorem, we can write

$$\tilde{I} = (\delta + Q) * I = \mathcal{F}^{-1}\big(\mathcal{F}(\delta + Q) \cdot \mathcal{F}(I)\big) \quad (36)$$

$$\Leftrightarrow I = \mathcal{F}^{-1}\big(\mathcal{F}(\tilde{I})/\mathcal{F}(\delta + Q)\big). \quad (37)$$

As $\mathcal{F}\delta \equiv 1$ and Q is small, we can safely divide by $\mathcal{F}(\delta + Q)$ in the Fourier space. Also, as Q is fixed, we only need to compute $1/(\mathcal{F}(\delta+Q))$ once and can then compensate arbitrary images by only calculating one forward Fourier transform, a per-element multiplication of the Fourier images and one backward Fourier transform. This can be done directly using complex images, since we always have two corresponding values per pixel.

6.4 Results

Fig. 8 summarizes the effect of compensating scattering effects on different kind of images. The resulting point spread function is shown in Fig. 9.

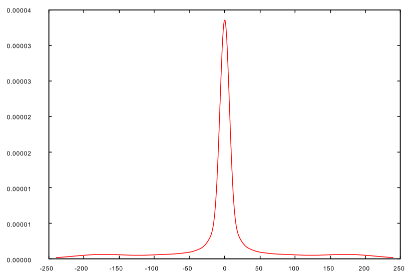
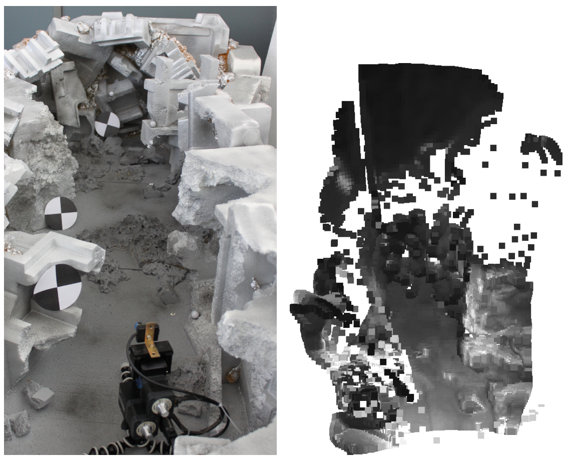


Figure 9: Point spread function Q as a function of the radius.

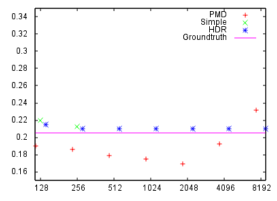


(a) Mockup with markers (b) 3D view

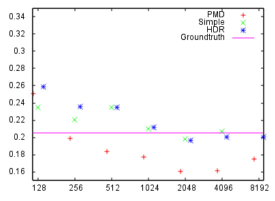
Figure 10: Mockup with markers

7 RESULTS

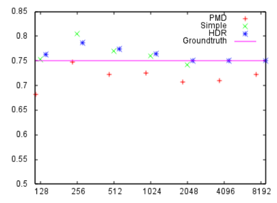
To verify our combined calibration efforts, we set up a Styrofoam mock-up (Fig. 10), motivated by an urban search and rescue scenario. We marked several spots inside the mock-up and measured



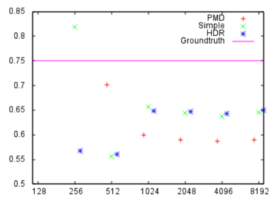
(a) White 205 mm



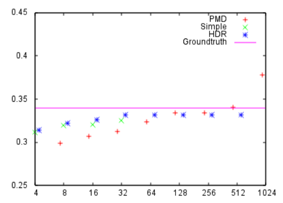
(b) Black 205 mm



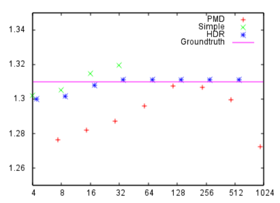
(c) White 750 mm



(d) Black 750 mm



(e) Reflector 340 mm



(f) Reflector 1310 mm

Figure 11: Distance measurements of pixels with hand-measured groundtruth. Green crosses are our inverse model applied to single measurements, blue stars are HDR recordings accumulated starting from 4 µs. Red pluses mark the values obtained from the manufacturer's driver and the violet bar is the hand-measured groundtruth for comparison.
On close ranges (a) or on highly reflective targets (e, f) single measurements quickly saturate, while the HDR model still works, ignoring saturated values. The manufacturer seems to model saturated values better, however the benefit of HDR recordings is apparent. The black are of the marker in (d) is overlayed by lens-scattering and multi path reflections.

the distance to the camera using a tape measure which we used as ground-truth for comparison.

We then made 64 recordings of the scene with different integration times from 8 to 8192 µs and we calculated the mean difference and standard deviation for several evaluation methods. The results are shown in Fig. 11

8 CONCLUSIONS AND FUTURE WORK

The main contribution of this paper is to provide a ToF sensor model which handles different integration times and thus is capable of producing HDR images. In contrast to previous calibration approaches we try to generate an accurate model from the beginning instead of adjusting the simplistic model given in (4). Being capable of recording HDR images we showed a straight-forward method to calibrate and compensate lens-scattering effects. We believe that exact HDR images are also required to reliably compensate effects of multi-path-reflections.

Furthermore, no extensive calibration equipment is required. In principle, a checkerboard and a white wall with self-printed markers is sufficient. However, some kind of automatically moving device (such as a PTU or a robot arm) helps to significantly reduce the manual work and increase the accuracy. We conclude by listing some phenomena left out in this paper, we esteem worthwhile investigating.

8.1 Temperature Compensation

It should be possible to take the temperature of the sensor and LED into account in the sensor model. As both temperatures are measured with an unknown delay, this also requires modeling how the actual sensor temperature changes given certain integration or waiting times and ambient temperature (which also usually is not known exactly). Furthermore, the temperature may have different effects on the generated light wave and the photon response curve. This is hard to determine since the temperature of both cannot be varied independently.

8.2 Ambient Light Effects

Fig. 12 shows how strong ambient light can have nontrivial effects on the generated signal. Partially, this is compensated by SBI (Möller et al., 2005, §4.2), but this compensation is not perfect. We left this calibration out, since pure ambient light is not directly measurable by our sensor and has very different effects on different pixels. Also our application will usually assume very little ambient light, which we assume to be negligible.

8.3 Multi-Path Reflections

Very complex problems arise if multi-path reflexions are to be considered. We think, essentially this can only be solved by applying an actual raycaster to a model of the scene and successively refine this model (Jimenez et al., 2012). The prob-

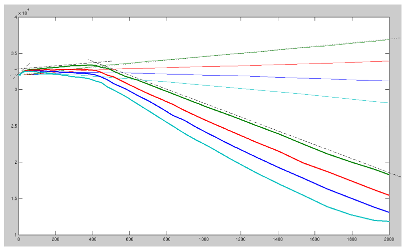


Figure 12: Raw values of PMD sensor, showing non-linearities and saturation effects. Each line are raw values of a single pixel in each sub-image. The right axis is the integration time in µs. Thick lines are with high back-light illumination, thin lines with almost no back-light. Notice that none of the thick curves is monotonic and all have substantial bends at approximately the same integration time. This behavior is different for different pixels and also depends on the temperature of the sensor.

lem gets arbitrary complex, if surfaces with non-Lambertian reflection (such as mirrors or retro-reflectors) or surfaces outside the camera's field of view have to be considered.

ACKNOWLEDGEMENTS

This work has partly been supported under DFG grant SFB/TR 8 "Spatial Cognition".

REFERENCES

Castaneda, V., Mateus, D., and Navab, N. (2011). SLAM combining ToF and high-resolution cameras. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 672–678.

Fuchs, S. and Hirzinger, G. (2008). Extrinsic and depth calibration of ToF-cameras. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–6.

Fuchs, S. and May, S. (2008). Calibration and registration for precise surface reconstruction with time-of-flight cameras. *Int. J. Intell. Syst. Technol. Appl.*, 5(3/4):274–284.

Hertzberg, C., Wagner, R., Frese, U., and Schröder, L. (2013). Integrating generic sensor fusion algorithms with sound state representation through encapsulation of manifolds. *Information Fusion*, 14(1):57–77.

Jimenez, D., Pizarro, D., Mazo, M., and Palazuelos, S. (2012). Modelling and correction of multipath interference in time of flight cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 893–900.

Kahlmann, T., Remendino, F., and Ingensand, H. (2006). Calibration for increased accuracy of the range imaging camera SwissRanger. In *TM Proceedings of the ISPRS Com. V Symposium*.

Lange, R. (2000). *3D Time-of-Flight Distance Measurement with Custom Solid-State Image Sensors in CMOS/CCD-Technology.* PhD thesis, University of Siegen.

Lindner, M. and Kolb, A. (2006). Lateral and depth calibration of PMD-distance sensors. In Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Nefian, A., Meenakshisundaram, G., Pascucci, V., Zara, J., Molineros, J., Theisel, H., and Malzbender, T., editors, *Advances in Visual Computing*, volume 4292 of *Lecture Notes in Computer Science*, pages 524–533. Springer Berlin Heidelberg.

May, S., Fuchs, S., Droeschel, D., Holz, D., and Nüchter, A. (2009). Robust 3D-mapping with time-of-flight cameras. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS'09, pages 1673–1678, Piscataway, NJ, USA. IEEE Press.

Möller, T., Kraft, H., Frey, J., Albrecht, M., and Lange, R. (2005). Robust 3d measurement with pmd sensors. In *In: Proceedings of the 1st Range Imaging Research Day at ETH*, pages 3–906467.

Mure-Dubois, J. and Hügli, H. (2007). Optimized scattering compensation for time-of-flight camera. *Proc. SPIE*, 6762:67620H–67620H–10.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. W. (2011). KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136.

OpenCV (2014). OpenCV. `http://opencv.org`.

PMDtechnologies (2012). pmd[vision] CamBoard nano. `http://www.pmdtec.com/products_services/reference_design.php`. last checked 2014-01-10.

Sabov, A. and Krüger, J. (2010). Identification and correction of flying pixels in range camera data. In *Proceedings of the 24th Spring Conference on Computer Graphics*, SCCG '08, pages 135–142, New York, NY, USA. ACM.

Weingarten, J., Gruener, G., and Siegwart, R. (2004). A state-of-the-art 3D sensor for robot navigation. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2155–2160 vol.3.