Delft University of Technology
Master's Thesis in Embedded Systems

Development of a Low-Cost, Solid-State, Line-Laser Distance Sensor

An Integrated Approach

H. van der Molen

Robot Security Systems

embedded software

TU Delft Delft University of Technology

Development of a Low-Cost, Solid-State, Line-Laser Distance Sensor: An Integrated Approach

Master's Thesis in Embedded Systems

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

H. van der Molen
H.vanderMolen-1@student.tudelft.nl

4th December 2017

Author
H. van der Molen (H.vanderMolen-1@student.tudelft.nl)
Title
Development of a Low-Cost, Solid-State, Line-Laser Distance Sensor:
An Integrated Approach
MSc presentation
December 14th, 2017 - 11:00AM

Graduation Committee

Prof. dr. K. G. Langendoen,	Delft University of Technology, EWI, Embedded Software
Prof. dr. ir. P. P. Jonker,	Delft University of Technology, 3ME, Cognitive Robotics
Prof. dr. ir. W.A. Serdijn,	Delft University of Technology, EWI, Bioelectronics
Drs. ing. T. J. Hijzen,	Robot Care Systems

Abstract

Many robotic systems rely on laser ranging sensors to navigate and map their environment. As robots are getting more advanced, smaller and cheaper, their sensors need to decrease in size and cost, while increasing their reliability and accuracy. Current laser-based sensors have difficulty to meet these properties. They are either large, expensive, contain moving parts or provide insufficient amounts of information.

This thesis introduces an integrated approach for the development of an eye-safe, low-cost, solid-state, wide-angle line-laser distance sensor, tackling these challenges. A prototype is derived by constructing a generic, camera independent, triangulation model. The model, combined with a set of pre-defined requirements, is used to select hardware components and predict the sensor limits. A unified calibration step is proposed to estimate both camera intrinsics as well as misalignment errors with the same set of data. Additionally, a microsecond-accurate open loop synchronization system for laser activation and imager exposure is presented.

Tests show that the prototyped sensor is able to measure distances up to 3 meters with an error of 4%. At 2 meter, the error is just under 2%. Additionally, the solid-state prototype has a field-of-view of 105 degrees, an angular resolution of 0.8 degrees, an update rate of 10Hz and an estimated cost of just below $35.

Contents

Chapter 1

Introduction

The market for consumer robotics is increasing at a fast pace with new drones, vacuum cleaners or other autonomous systems popping up in rapid succession. Advances in technology allow the miniaturization of components, decreasing its costs while increasing the computational power and performance. As consumers get accustomed to these improvements, they also start to expect a more genuine understanding and interaction between the robot and the environment [39], putting additional pressure on the development of robots.

One of the key tasks for autonomous mobile robots is navigation and (self-)localization. To enable a robot to navigate itself in an environment, it needs to be able to get information on where objects are located. If the robot also needs to localize itself, the retrieved information has to be detailed enough so that the robot is able to deduce key-points or landmarks from its observations. Combined, these constraints produce the following properties of appropriate distance sensors:

A sensor suitable for navigation and localization needs to be able to *observe a wide area of space* or *large field-of-view* in which it can describe the position of several obstacles. In other words, the sensor needs to be able to estimate the *distance* towards objects. Additionally, a suitable sensor needs to be *compact and low-cost* to allow a decrease in size and cost of the robot. Most preferably such a sensor also lacks moving parts and hence is *solid-state*, minimising (mis)alignment errors or inaccuracies, as there are no gears or movings components which can wear down.

Table 1.1 shows a high-level comparison of different kind of distance estimation sensors. One of the most basic sensors is the ultrasound sensor. It works on the principle of echo-location of sound waves. While low-cost and compact, they tend to have low resolution and cannot deliver the required details for localization. Radar works on the same principle, except it utilizes radio waves instead of sound. Stereo Vision sensors (consisting of multiple cameras facing the same direction) may provide the appropriate resolution for an autonomous robot, but they generally require complicated and intensive computations and image processing, increases its cost. Laser-based systems, on the other hand, have high resolution, large field-of-view, are relatively compact, but may lack a solid-state design or ignore the low-cost property. Because of their effectiveness, many

	Ultrasound	Radar	Stereo Vision	Laser
Field of View	-	+	+	+
Compact and Low-cost	+	-	-	+/-
Solid-State	+	+	+	+/-
Resolution	-	-	+	+

Table 1.1: High level comparison of common used distance estimation sensors.

attempts have been made to solve these last two properties [5, 6, 37, 44, 33, 29, 29, 17, 38]. Nonetheless, none of these have sufficiently succeeded in creating a generic model, fulfilling the previously stated properties. This thesis is focussed on the development and prototyping of such a model, following an integrated approach.

The general idea of the construction of a low-cost laser distance sensor is to use a triangulation setup as shown in Figure 1.1:left. It utilizes a camera and laser, placed at known distance and angle from each other. In this setup, the laser sends a pulse of light, which is reflected by an object and then captured by the pixels of the imager in the camera. Depending on the distance of the object, the reflection of the laser is observed at a different pixel position. Using the known triangulation configuration, this pixel position eventually can be translated into a distance estimation.

As a single laser pulse results in a very small field-of-view, additional steps have to be taken to ensure the observability of a larger area. Typically, the laser-camera system is mounted on a rotating disk. By controlling the rotation of the disk, precise estimates can be made at different angles, resulting in an increased field-of-view. This approach, however, violates the solid-state property. To hold this property, a line laser can be utilized. A line-laser is a laser pointer which is extended with a lens. The added lens diffracts the laser light such that the laser projects a line instead of a point. By observing the full width of the line with a camera, distance estimations across the field-of-view of the camera can be made. Figure 1.1:right shows a rendering of the, in this thesis deduced, prototype.

In addition to the low-cost and solid-state properties, the presented prototype should meet the following requirements:

1. Solid-state with a cost of $<$ \$30 each, when producing 1000 pieces.
2. Dimension should be within 10x3x4cm.
3. $>$ 2 meter detection range with a resolution $<$ 1%.
4. Planer, horizontal field-of-view of 120 degrees with a resolution of $<$ 1 degree.
5. $>$ 10Hz update rate. That is, 10 times a full 120-degree scan per second.
6. Eye-safe, Class 1 Laser product according to the NEN IEC 60825-1:2014 [24]
7. Fully embedded processing.

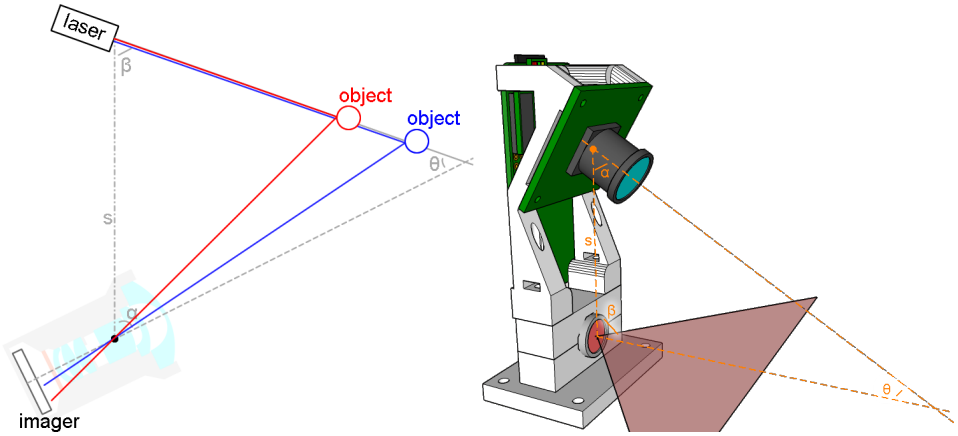


Figure 1.1: *Left:* General idea of a triangulation laser-distance sensor. A laser (top) sends a pulse and depending on the position of an object, the reflection is observed at different pixel location in the camera. At the bottom a cross-section of a camera is shown, containing the lens and imager. *Right:* Rendering of the prototype presented in this thesis, utilising a line laser (bottom) and camera (top).

To ensure that the requirements are met as closely as possible, an integrated approach is taken for the development of the prototype. That is, models, parts, and settings are developed, derived and selected in such a way that the combined performance ensures minimal performance-loss. As a result, the requirements have resulted in several contributions. Most notably are:

1. Extension on the current laser-triangulation models, allowing the use of wide-angle (high distortion) lenses and line-lasers.

2. Development of tooling allowing (theorized) limit estimation of a prototype and selecting optimal triangulation parameters.

3. Integrated calibration process to estimate both the intrinsic camera and the triangulation parameters.

4. Open-loop synchronization of the camera exposure and laser activation.

5. A prototyped and verified, low-cost, solid-state, line-laser distance sensor.

These contributions are presented in the next chapters. The first chapter (Chapter 2) introduces a comparison of existing state-of-the-art systems, their performance, models, and limitations. These models are then extended to work with high-distortion lenses and line-lasers in Chapter 3. Additionally, Chapter 3 provides the computations and derivations to estimate the (theorized) limitations of a given setup. This assumes a general knowledge of lens distortion, projection and imager properties, for which a refresher is provided in Chapter A. Using the presented model, Chapter 4 presents the

constructed prototype and its limitations. Details on the hardware selection are given in Chapter B. This thesis moves from the hardware to the software domain in Chapter 5, which presents the laser-camera synchronization. Finally, the image processing and distance estimation results are provided in Chapter 6. The conclusion and future research are presented in Chapter 7. Additional remarks, proofs and derivations are given the remaining appendices: Chapter A-Chapter D.

Chapter 2

Existing Systems

A wide variety of laser distance sensing systems exists. This chapter provides an analysis and comparison of the physical limits and properties of the state-of-the-art production- and research- sensors. Additionally, the theory and limitations of existing triangulation models are presented.

2.1 State of the art performance

By utilizing the focused beam of light of a laser, precise distance estimates can be made over large distances. For example, the Velodyne HDL-64E [7] is able to measure distances up to 120 meters and the Luminar LiDAR is claimed to be able to make distance estimates up to 250 meters [32]. Unfortunately, as these sensors are high-performance systems, their prices are in the range of several tens-of-thousands of dollars. Therefore such sensors are not very friendly to the consumer robotics market. Existing middle-class systems such as the Sick LMS111 [8] and Hokuyo UTM-30LX [13] are compact and able to measure distances up to 30-50 meters with several millimeters of accuracy, yet have still price tags of several thousands of dollars. The current available low-end sensors come in at hundreds of dollars. Among these sensors are the RPLidar [5] and Sweep [6], costing around 350-450 dollar each. The first is claimed to be able to measure distances up to 6m with $< 1.5\%$ accuracy whereas the latter can measure up to 40 meters, but with high error ($> 5\%$) at short distances.

All these sensors use rotating parts for measurement and therefore may be prone to errors due to movement, stutter or wear down of gears. Moving parts also increase the mechanical complexity of a sensor, hence research is moving towards the development of solid-state systems. MIT is, for example, working with DARPA to produce a small (0.5x6 millimeter) single-chip solution [37]. It contains both sending and receiving drivers and is claimed to be able to scan an area of 51 degrees wide with a distance up to 2 meters with $< 2\%$ resolution. Another system is being developed by Quanergy [44]. They hope to start the production of their solid-state S3 3D sensor early 2018. The S3 is claimed to measure distances up to 100 meters with 0.1% accuracy and a horizontal and vertical field-of-view of 120 degrees. Unfortunately, as these solid-state approaches depend on

the use of specialized directional antennas and phased arrays, they currently have high hardware costs. With the technology still being developed, low-cost consumer friendly variants are not to be expected anytime soon and may take several years to come into existence.

Low-cost solid-state systems, have been constructed by configuring (line) lasers and light sensitive parts in a triangulation setup [19, 33, 35, 38]. In such a setup both laser and light-sensitive part face the same direction in a known configuration. A comparison of these systems with current low-end rotating sensors is given in Table 2.1

When comparing these setups, a wide variation in distances and resolutions can be observed. Not one sensor is close to the claimed properties of the solid-state directional antenna approaches, yet the results and price of the triangulations setups are well suited for many robotic applications. Interestingly, none of the triangulation setups are able to properly detect a signal further than 6 meters: a limit as a result of the maximum allowable laser output for eye-safety and the amount of background irradiance produced by the sun.

If it is assumed that all requirements for our prototype will be met, the presented prototype would exhibit the following properties:

- *Price:* $< \$30$ (when constructing 1000 units)
- *Field-of-view:* 120 degrees
- *Angular Resolution:* < 0.9 degrees
- *Max distance:* $>$ 2meter
- *Resolution @ 2m* $< 1\%$
- *Speed* $>$ 10Hz
- *Samples* $>$ 1333/s
- *Dimensions* 100 x 30 x 40 mm

2.2 Triangulation

The presented triangulation systems all use slightly different setups, yet they can be generalized in the setup shown in Figure 2.1.

The generalised model is an adaption of the model presented by Konolige et al [29], which states that the distance towards an object can be computed by:

$$q = \frac{s * f}{x} \tag{2.1}$$

In this equation, q and s are the distance-to-object and distance-between-laser-camera respectively. The variables f and x represent the focal point of the lens (the distance

Sensor	RPLidar [5]	Sweep [6]	Nexus 6 [19]	Pentium II [33]	Theoretical [35]	Mini robot [38]
Type	Rotating	Rotating	Fixed	Fixed	Fixed	Fixed
Laser	Point	Point	Line	Line	Line	Line
Method	Triangulation	Time-of-Flight	Triangulation	Triangulation	Triangulation	Triangulation
Price	$450	$350	$49 (+$400)	$600	-	$60
Field-of-view	360 deg	360deg	60 deg	55 deg	16	50 deg
Angular Resolution	0.9 deg	< 0.5 deg	< 0.11 deg	< 0.49 deg	< 0.01 deg	< 0.1 deg
Max distance	6m (1%)	40m (1 - 18cm)	5.8m (13.6cm)	5.2m	3m	0.6m (<2cm)
Min distance	0.15m (0.5mm)	0.3m (25%)	2.7cm	-	0.32	-
Resolution @ 2m	<1%	<1%	9.1cm	2.6cm	9.3cm	<2cm
Speed	3600 deg/s	3600 deg/s	30Hz	30Hz	-	30Hz
Samples	4000/s	1000/s	14400/s	14820/s	-	14400
Dimensions (mm)	76 x 76 x 41	65 x 65 x 62	159 x 83 x 50	5000 x 150 x 100	-	< 100 x 100 x 100
Weight	340g	120g	>184g	-	-	-

Table 2.1: Comparison of different low-cost laser distance sensors. The first two sensors are commercially available. The third (Nexus 6) is a smartphone based sensor and the fourth (Pentium II) is a large test-rig with NTSC camera. The fifth sensor is a theoretical evaluation, included to compare the limitations of measurements and the final sensor is integrated into a miniature robot of 10x10x10 cm.

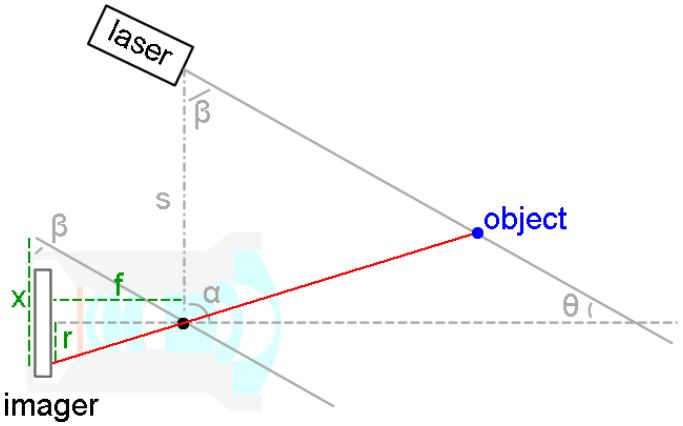


Figure 2.1: Generalised triangulation setup based.

imager-optical center) and the difference between the pixel position of the reflected laser and the ray through the optical center, parallel to the laser.

While it can be used to compute an object-distance from a pixel coordinate, the model relies on several important assumptions.

Foremost, is assumes that the angle α equals 90 degrees, ensuring that s is perpendicular to the optical axis. Releasing this assumption can be done by generalising (2.1) as shown in Chapter D.1. The generalised formula reads:

$$q = s * \frac{sin(\beta)sin(\alpha) - y_n * sin(\beta)cos(\alpha)}{sin(\theta) + y_n * cos(\theta)}. \tag{2.2}$$

With $y_n = \frac{r}{f}$.

Secondly, the simplified model (and its generalized version) both have a direct link between the focal point of the lens and the estimated distance. Therefore, neither of these equations allow high amounts of (nonlinear) lens distortion, hence they cannot be used with wide-angle or fisheye lenses.[1]

Lastly, when a line laser is used, both models assume that the same triangulation parameters hold across all pixels in the image. This, however, is not the case when there are slight rotational or translational misalignments. When not accounted for, these small misalignments can result in significant errors at larger distances.

In the next chapter a new model is presented, able to handle such misalignments and nonlinear distorted lenses.

[1]It should, however, be noted that y_n in the generalised equation (2.2) describes the projection of an undistorted lens (Chapter A). When the distortion model of Heikkila [21] is used, the vertical projection of an undistorted, normalized pixel can directly be connected with y_n, allowing Equation (2.2) to be used with a wide variety of lenses.

Chapter 3

Sensor Model

As shown in Chapter 2, existing models are not able to properly combine wide-angle lenses, (line) lasers and misalignment errors in a triangulation setup. This chapter introduces a new and generic model, which is able to handle such cases.

First, a description of a lens distortion model is given, able to handle wide-angle lenses [22, 28, 12]. This is followed by the introduction of the newly derived triangulation model. Additionally, the math and models from the IEC 60825 standard [24] for eye-safety are introduced, together with background irradiance estimations from the Spectra 1.5 standard [10]. The chapter is concluded with an overview of all parameters in the full model.

3.1 Image projection and distortion

Large amounts of literature have been dedicated to describing the translation from 3D world coordinates to 2D pixel positions and vice versa [28, 34, 41, 22, 43, 16]. For this thesis the projection and distortion models presented by Heikkila [22] and Kannala [28] are used. These models are also used in well-known tools such as the OpenCV vision library [42] and the Caltech Camera Calibration Toolbox for Matlab [12]. As their details have been extensively covered by different papers, this thesis will only present the models as a set of equations without deeper analysis of correctness.

3.1.1 Forward projection: 3D to 2D

The most simple transformation from a 3D world coordinate, $p = [x\ y\ z]^T$, to a 2D pixel position, $p_p = [x_p\ y_p]^T$, is by using the Pinhole Camera model [41]. This model assumes no lens, hence the projection transformation is written as:

$$p_p = \frac{1}{z}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \tag{3.1}$$

Where p equals the 3D world coordinate of an object, and p_p the corresponding 2D pixel position. As p_p in this case equals p, normalised over the z-axis (depth), p_p also equals the normalised 3D world coordinates, $p_p = p_n = [x_n\ y_n\ 1]^T$.

When a lens is added, the computation to determine the pixel position is extended by including the Camera Matrix, KK [41]:

$$p_p = KK * p_n \Rightarrow \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \tag{3.2}$$

Where f_x and f_y are the focal distances, s the skew in the imager (that is: when pixels are not perpendicular aligned) and (c_x, c_y) the pixel position of the optical centre. This extension assumes that a *perfect lens* is used: that is, no distortion is introduced and the imager and lens are aligned perpendicular to each other.

To handle distortion, an additional step needs to be added. By computing p_p via p_n, we can handle distortion, δ_d, as follow [28]:

$$p_p = KK * (p_n + \delta_d) = KK * p_d \tag{3.3}$$

Where the distorted 2D position, p_d, is computed as:

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} + \begin{bmatrix} r^2 x_n & r^4 x_n & r^6 x_n \\ r^2 y_n & r^4 y_n & r^6 y_n \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} + \begin{bmatrix} 2x_n y_n & r^2 + 2x_n^2 \\ r^2 + 2y_n^2 & 2x_n y_n \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \tag{3.4}$$

With $r = \sqrt{x_n^2 + y_n^2}$, $[k_1\ k_2\ k_3]^T$ representing the lens distortion parameters and $[p_1\ p_2]^T$ the tangential distortion[1].

3.1.2 Backward projection: 2D to 3D

As the laser distance sensor does not need to compute the 3D to 2D pixel transformation, but rather requires the inverse computation, the backward projection of the model is of more importance. Due too the usage of a non-linear distortion model, this computation cannot derive an exact projection. Additionally, as the forward projection eliminates a dimension, only the normalized position, p_n, can be estimated from p_p.

To compute p_n from p_p, the inverse of the non-linear distortion model needs to be computed. Heikkila [21] proposed an iterative process, which is adapted in this thesis. The proposed approach starts by inverting the computation of the Camera Matrix, KK:

$$y_p = f_y * y_d + c_y \qquad \Rightarrow y_d = \frac{1}{f_y} * (y_p - c_y) \tag{3.5}$$

$$x_p = f_x * x_d + s * y_d + c_x \qquad \Rightarrow x_d = \frac{1}{f_x} * (x_p - c_x - s * y_d) \tag{3.6}$$

[1]The latter is the induced error when the imager and lens are not perfectly perpendicular to each other.

	Description
s	Distance between the laser and the optical centre of the camera.
q	Object-sensor distance, measured perpendicular to s.
α	Rotation of the camera with respect to s.
β	Rotation of the laser with respect to s.
θ	Subtended angle of the central axis of the laser with the optical axis of the lens.
ψ	Rotation of the laser over its central axis. This rotation is also called the laser-roll.

Table 3.1: Triangulation parameters and their description, used in the proposed model.

Next, an approximation of p_n, called $\tilde{p_n}$, is computed by repeated executing of the following steps:

$$\tilde{p}_n = \begin{bmatrix} \tilde{x}_n \\ \tilde{y}_n \end{bmatrix} = \begin{bmatrix} x_d \\ y_d \end{bmatrix} \quad \text{Initialisation} \tag{3.7}$$

$$\tilde{r} = \sqrt{\tilde{x}_n^2 * \tilde{y}_n^2} \quad \text{step 1} \tag{3.8}$$

$$d_r = 1 + \tilde{r}^2 k_1 + \tilde{r}^4 k_2 + \tilde{r}^6 k_3 \quad \text{step 2: radial distortion} \tag{3.9}$$

$$d_t = \begin{bmatrix} 2\tilde{x}_n\tilde{y}_n & \tilde{r}^2 + 2\tilde{x}_n^2 \\ \tilde{r}^2 + 2\tilde{y}_n^2 & 2\tilde{x}_n\tilde{y}_n \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \quad \text{step 3: tangential distortion} \tag{3.10}$$

$$\tilde{p}_n = \frac{1}{d_r} \left(\begin{bmatrix} x_d \\ y_d \end{bmatrix} - d_t \right) \quad \text{step 4: correction, continue step 1} \tag{3.11}$$

Increasing the number of iterations improves the projection results. The OpenCV library uses 20 iterations, whereas the Caltech Toolbox only 5. In this thesis, 5 iterations will be used.

3.2 Triangulation

Using the backward-camera projection, presented in the previous section, a pixel position can be translated to a normalised 2D coordinate, $p_n = [x_n \; y_n]^T$. This section shows how these 2D coordinates can be used to compute the corresponding 3D position, and hence estimate a distance, by using a known triangulation configuration.

3.2.1 An ideal setup

A schematic 3D triangulation setup is shown in Figure 3.1. For this setup the same naming conventions as used by Konolige et. al [29] are used with the addition of several other parameters. An overview of all parameters is displayed in Table 3.1.

Using these definitions and the normalised, undistorted pixel coordinates, the z-

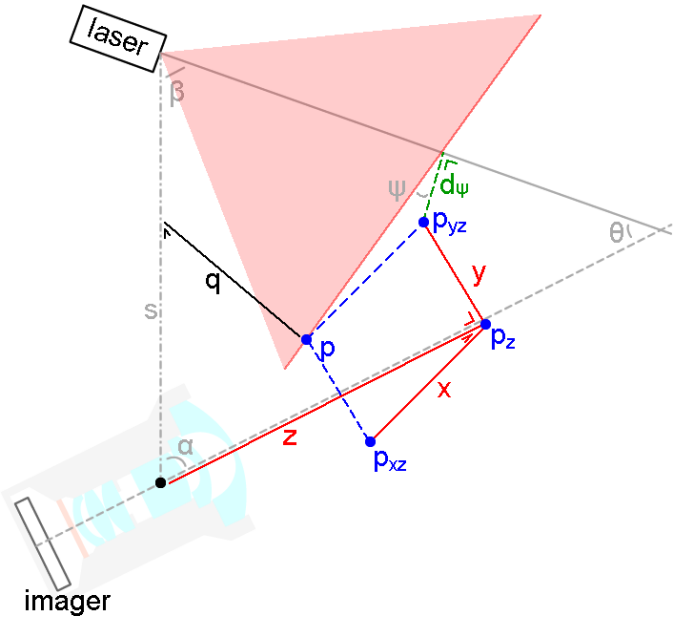


Figure 3.1: The 3D line-laser triangulation model used in this thesis. This image is a side view of the setup, displaying the ZY-plane, with the X-axis pointing outwards. Note that the line laser is aligned in the same direction, with ψ, representing the roll of the laser in its central axis. In an ideal setup, ψ equals 90 degrees, aligning the projected laser line horizontally with the imager of the camera. The purpose of a triangulation setup is to derive the distance q towards a point p, using p_n (the normalised version of p) and a known triangulation configuration $(\alpha, \beta, \theta, \psi, s)$.

position in the camera reference frame can be computed with:

$$z = s * \frac{sin(\beta)}{sin(\theta) + y_n * cos(\theta) + \frac{x_n}{tan(\psi)}} \tag{3.12}$$

Which allows the estimation of the 3D position, p:

$$p = z * p_n = z * \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \tag{3.13}$$

The distance towards an object, q, is then computed with:

$$q = z * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2 + x_n^2} \tag{3.14}$$

A full derivation of these equations is shown in Chapter D.1.

While the presented model allows the estimation of the 3D coordinate of a pixel position, its computations are based on an ideal and perfectly aligned setup. It assumes that the exact (relative) positions and orientations of both laser and camera are known. Physical systems commonly exhibit small misalignments, hence the shown computations can only be used as a theoretic model for designing a triangulation system.

3.2.2 Handling misalignments

To address misalignments in the presented model, a more practical approach needs to be taken. When looking at the setup shown in Figure 3.1, it can be reasoned that the projected laser line can be described as a 3D plane. As each point in this plane correlates with a possible observable 3D position, it can be used to translate p_n to p.

In order to make this translation, we start with the geometrical description of a 3D plane:

$$a * x + b * y + c * z + d \tag{3.15}$$

Which, after rewriting, results in:

$$z = \frac{a * x + b * y + d}{-c} = a_c * x + b_c * y + c_c \tag{3.16}$$

Where (a_c, b_c, c_c) are the plane parameters. Mapping these parameters to Figure 3.1, we can state that they describe the laser-plane with respect to the optical centre of the camera. In an ideal setup, these plane-parameters can be used to compute the triangulation parameters:

$$\theta = tan^{-1}\left(\frac{1}{b_c}\right) \qquad \psi = 0.5 * \pi - tan^{-1}(-a_c * cos(\theta))$$
$$\alpha = \pi - (\beta + \theta) \qquad s = \frac{sin(\theta) * c_c}{sin(\beta)}$$

These parameters can also be used to compute the z-position of a normalised point:

$$z = \frac{c_c}{1 - (a_c * x_n + b_c * y_n)} \tag{3.17}$$

Which eventually allows the computation of the 3D world position as shown by (3.13). It should also be noted that this equation can be used as a substitute for (3.12), which was presented in the previous section.

As the laser-plane parameters can be estimated with the capturing of several frames (which will be shown in Chapter 4), it bypasses the requirement of determining the exact values of the triangulation configuration and hence allows handling of misalignment errors.

All derivations proofs for the presented equations can be found in Chapter D.

3.3 Laser safety

Each system including a laser is required to be validated and labeled to ensure that a user can take the proper precautions to use the product safely. Labelling of a laser product is done by classifying it into a specific class, using the equations and rules set by the IEC 60825 [24] standard. A brief summary of the available laser-product classes is shown in Table 3.2.

The IEC 60825 states that if the outputted energy of a system, able to reach the eye or skin, is below a specified Accessible Emission Limit (AEL), the system can be labeled as a Class 1 laser product. In the next sections, the AEL computation for a laser system utilizing a laser with a wavelength in the 600-1050nm range is shown. In this range the laser light is more dangerous to the eye then skin, therefore, only the ocular limits are presented. In addition to the AEL, the computations to determine the maximum allowable laser power and Class 1 safety operating area (Nominal Ocular Hazard Distance, NOHD) are given.

3.3.1 Accessible Emission Limit

The AEL is depending on several factors, but, assuming a single laser source with a wavelength between 600 and 1050 nm, producing pulses with a length between $5 * 10^{-6}$ and 1/15 seconds, the following equation holds:

$$AEL = 7 * 10^{-4} * t^{0.75} C_4 C_6 \tag{3.18}$$

In which the AEL is the maximum allowable energy in Joule and C_4 and C_6 are correction-factors to include the effect of different wavelengths or thermal damage respectively.

Class	Description
1	Eye-safe under all operating conditions. The produced illumination can be both visible and invisible.
1M	Eye-safe under all operating conditions, unless it is observed with optical instruments. The produced illumination can be both visible and invisible.
2	Eye-safe unless more than 0.25s (natural aversion response) is stared into the beam. The produced illumination is solely visible but might include invisible light under certain conditions.
2M	Eye-safe unless more than 0.25s (natural aversion response) is stared into the beam or when it is observed with optical instruments. The produced illumination is solely visible but might include invisible light under certain conditions.
3A	Potentially hazardous. The limits of this class are five times the limits of class 1 and 2.
3B	Likely to be hazardous for eye or skin, but the diffuse reflection is safe. A continuous source may not exceed a maximum output of 500mW.
4	Hazardous to eye or skin when using direct light and potentially hazardous when using indirect light. Might set materials on fire.

Table 3.2: Laser safety classes as defined by the IEC 60825 [24] standard

C_4 and C_6 are determined by:

$$C_4 = \begin{cases} 1 & \text{for } 600 \leq \lambda < 700nm \\ 10^{0.002(\lambda - 700)} & \text{for } 700 \leq \lambda < 1050nm \end{cases} \tag{3.19}$$

$$C_6 = \begin{cases} 1 & \text{for } \alpha \leq \alpha_{min} \\ \alpha/\alpha_{min} & \text{for } \alpha_{min} < \alpha \leq \alpha_{max} \\ \alpha_{max}/\alpha_{min} & \text{for } \alpha > \alpha_{max} \end{cases} \tag{3.20}$$

Where λ represents the wavelength in nanometers and α a correction factor based on the angle of acceptance of the setup. The angle of acceptance is the subtended angle at a point in space from which the laser source is observed. It is computed in milliradians and depends on the diameter of the laser source, d (in mm), and the distance, $r >= 100$mm, at which the laser is observed. The maximum angle can therefore be computed as:

$$\alpha = 2000 * tan^{-1}\left(\frac{0.5d}{r}\right) \Rightarrow 2000 * tan^{-1}\left(\frac{0.5d}{100}\right) \tag{3.21}$$

In addition to α, C_6 also depends on a minimum, α_{min}, and maximum, α_{max}, subtense. Both are expressed in milliradians and only depend on the duration time, t, of a pulse:

$$\alpha_{min} = 1.5 \tag{3.22}$$

$$\alpha_{max} = \begin{cases} 5 & \text{for } t < 625\mu s \\ 200 * t^{0.5} & \text{for } 625\mu s \leq t \leq 0.25s \\ 100 & \text{for } t > 0.25s \end{cases} \tag{3.23}$$

The above equations compute in essence the maximum amount of allowable energy in a single pulse. When a system produces several pulses, additional checks need to be performed. For a pulsing Class 1 system, the following conditions should hold:

1. The exposure of a single pulse, AEL_{single}, should be below the AEL.
2. The average power of all pulses in a time window T, $AEL_{avg.T}$, should be below the corresponding power of the AEL of a single pulse of a duration T, AEL_T.
3. For thermal safety, the energy per pulse, $AEL_{s.p.train}$, should be below the energy of a single pulse, AEL_{single}, multiplied with a parameter C_5.

Or, in other words, the maximum allowable amount of energy of a single pulse for Class 1 classification is determined with:

$$min\left(AEL_{single}, AEL_T, AEL_{s.p.train}\right) \tag{3.24}$$

The following paragraphs describe each condition in more detail.

Condition 1 Typically a laser is stated to have a certain amount of output power, P_{laser} in Watts (J/s). Multiplying with the pulse duration, t, provides the total amount of energy (J) in a pulse. Assuming that all energy reaches the eye, we get:

$$AEL_{single} = P_{laser} * t \tag{3.25}$$

Hence, the limit of a single pulse can be expressed as:

$$AEL_{single} < AEL \tag{3.26}$$
$$AEL_{single} < 7 * 10^{-4} * t^{0.75} C_4 C_6 \tag{3.27}$$

Condition 2 Different wavelengths require different time-windows to analyze. The IEC 60825 specifies that in case of a Class 1 system with $\lambda > 400nm$ and no intentional long-term viewing into the laser source, a window of $T = 100$ seconds should be sufficient.

For determining $AEL_{avg.T}$, the number of pulses, N, within the window T needs to be computed, giving:

$$AEL_{avg.T} = AEL_{single} * N \tag{3.28}$$

For determining AEL_T, the same parameters as for the computation of AEL should be used, except $t = T$. Hence the full condition can be expressed as:

$$AEL_{avg.T} < AEL_T \tag{3.29}$$
$$AEL_{single} * N < 7 * 10^{-4} * T^{0.75} C_4 C_6 \tag{3.30}$$
$$AEL_{single} < \frac{7 * 10^{-4} * T^{0.75} C_4 C_6}{N} \tag{3.31}$$

Condition 3 In essence, this condition states that a single pulse multiplied with a thermal parameter should be below the computed AEL:

$$AEL_{s.p.train} < AEL \tag{3.32}$$
$$AEL_{single} * C_5 < 7 * 10^{-4} * t^{0.75} C_4 C_6 \tag{3.33}$$
$$AEL_{single} < \frac{7 * 10^{-4} * t^{0.75} C_4 C_6}{C_5} \tag{3.34}$$

In which C_5 is determined by:

$$C_5 = \begin{cases} 1 & \text{for } \alpha \leq 5 \text{ OR } \alpha > 100 \\ N^{-0.25} & \text{for } 5 < \alpha \leq \alpha_{max} \text{ AND } N \leq 40 \\ 0.4 & \text{for } 5 < \alpha \leq \alpha_{max} \text{ AND } N > 40 \\ N^{-0.25} & \text{for } \alpha > \alpha_{max} \text{ AND } N \leq 625 \\ 0.2 & \text{for } \alpha > \alpha_{max} \text{ AND } N > 625 \end{cases} \tag{3.35}$$

Where N is the number of pulses within a time window of T_2 seconds:

$$T_2 = \begin{cases} 10 & \text{for } \alpha \leq \alpha_{min} \\ 10 * 10^{(\frac{\alpha - \alpha_{min}}{98.5})} & \text{for } \alpha_{min} < \alpha \leq 100 \\ 100 & \text{for } \alpha > 100 \end{cases} \tag{3.36}$$

This computation requires the additional note that if multiple pulses occur within a window of T_i seconds, they are counted as a single pulse. For $\lambda \in [400, 1050]$, this window is defined as:

$$T_i = 5 * 10^{-6} \tag{3.37}$$

3.3.2 Nominal Ocular Hazard Distance

While the AEL computes the classification and the maximum allowable amount of energy to reach the eye, it does not mention safety distances nor includes computations regarding the shape of a laser beam. To this extent, the Nominal Ocular Hazard Distance (NOHD) and Maximum Permissible Energy (MPE) have been introduced.

In essence, the MPE of a laser system equals the AEL, only rewritten to J/m^2 (or J/cm^2) by including the surface of the pupil of an eye (A_{eye}). The IEC 60825 assumes an average diameter of 0.7 cm for the pupil, hence the MPE (J/cm^2) of a Class 1 system, under previously defined constraints is given by:

$$A_{eye} = \pi * (0.5 * d_{eye})^2 = \pi * 0.35^2 \sim 0.385 \tag{3.38}$$

$$H_{mpe} = \frac{AEL}{A_{eye}} = \frac{7 * 10^{-4} * t^{0.75} C_4 C_6}{0.385} = 18 * 10^{-4} * t^{0.75} C_4 C_6 \tag{3.39}$$

The NOHD of a system is the distance towards the laser source at which the average energy per surface-units equals the MPE. If the user cannot view the laser source from within the NOHD, the system is classified as safe and labeled as a Class 1 product.

Depending on the shape of the laser beam different computations need to be performed. The next two paragraphs show the steps to compute the NOHD given the known output power of a laser and the AEL. They also show the steps to determine the maximum allowable output power of a laser, $\max(P_{laser})$ given an NOHD and the AEL respectively. Both assume a simplified model of a line laser as the horizontal divergence of the beam is commonly several orders larger as the vertical divergence (radians vs milliradians). In addition, the assumption is made that the projected line has a uniform intensity.

NOHD from P_{laser} and AEL Computing the NOHD is done in two steps. First, the area of the projection of the laser, A_{laser}, is computed at the point where the average intensity equals the MPE :

$$A_{laser} = H_{mpe} * P_{laser} * t \tag{3.40}$$

Where P_{laser} is the output power of the laser in Watts (J/s) and t the pulse duration.

Assuming a rectangular projection with height y (cm) and a width determined by a horizontal divergence ϕ (radians), the NOHD (cm) is determined by:

$$nohd = \frac{A_{laser}}{2 * tan(0.5\phi) * y} \tag{3.41}$$

$$= \frac{H_{mpe} * P_{laser} * t}{2 * tan(0.5\phi) * y} \tag{3.42}$$

max(P_{laser}) from known NOHD and AEL When the NOHD is the limiting factor instead of the output power of the laser, a different approach needs to be taken. In such a case, the restricting NOHD is first used to compute the area of the laser projection at the given distance:

$$A_{laser} = 2 * nohd * tan(0.5\phi) * y \tag{3.43}$$

Which in turn can be used to compute the maximum allowable output power:

$$P_{laser} = \frac{A_{laser} * H_{mpe}}{t} \tag{3.44}$$

$$= 2 * nohd * tan(0.5\phi) * y * \frac{H_{mpe}}{t} \tag{3.45}$$

3.4 Background irradiance

Using the laser safety rules, the maximum output power of the laser can be determined. In order to make some estimations of the limitations of the prototyped sensor, we also need to know the amount of background irradiance produced by the sun. To this extent the Spectra 1.5 tables [10] are used. The Spectra tabels model the solar irradiance at different atmospheric conditions. It specifies the background irradiance in $W/m^2/nm$ per wavelength.

Limiting the amount of background irradiance will result in better detection of the laser reflection. As a laser only emits irradiance at a fixed wavelength, an optical bandpass filter is commonly used to block unwanted irradiance from the imager. Using the Spectra 1.5 tables and the width of the bandpass filter, an estimate of the total amount of background irradiance, P_{sun} in W/nm^2, can be made.

Assuming the simplified line-laser model of the previous section and a laser output, P_{laser}, the distance, x, at which the laser reflection is pp percent of the background irradiance, is computed with:

$$x = \frac{P_{laser}}{pp * P_{sun} * 2 * tan(0.5\phi) * y} \tag{3.46}$$

Where $pp \in (0, 1]$ with $pp = 1$ equalling 100 percent and y representing the height of the laser reflection.

3.5 Combined model

Combining all equations presented in the previous sections, result in a full model, able to estimate the distance sensor limitations and optimal configurations. In Table 3.3 a list with names and descriptions of all input-parameters of the complete model is shown.

Using this list of parameters, the model is able to estimate the sensor limitations, by executing the following steps sequentially:

1. The model starts by computing a vector of 2D pixel positions, $p_p = [p_x\ p_y]^T$, using the defined imager dimensions.
2. These pixel positions are then transformed into undistorted, normalised positions: $p_n = [x_n\ y_n\ 1]^T$ with the camera backward projection
3. Next, the triangulation parameters (α, β, ψ, s) are transformed into the laser-plane parameters (a_c, b_c, c_c).
4. Combining the laser-plane parameters and normalized positions, a distance estimation, q, is computed for each pixel position.
5. Having computed the distance for all pixels, limitations can be derived:
   - Per pixel the resolution of q can be estimated by computing the difference between consecutive pixels.
   - Per column of the imager, the maximum and minimum distance can be found.
6. Finally, the set of laser-parameters is used to determine the NOHD, maximum allowable laser power and detection distance with respect to the background irradiance.

The presented equations and steps have resulted in the build of a Matlab-tool to find the optimal triangulation values and determine the limitations of the prototype, as will be shown in the next chapters.

	Description
$[I_x\ I_y]$	Number of pixels in the imager: width and height
KK	Camera matrix, consisting of (f_x, f_y), (c_x, c_y) and $skew$
$[k_1\ k_2\ k_3]$	Lens distortion parameters
$[p_1\ p_2]$	Tangential distortion paramaters
s	Distance between the laser and the optical centre of the camera.
α	Rotation of the camera with respect to s.
β	Rotation of the laser with respect to s.
ψ	Rotation of the laser over its central axis.
d	Laser diameter
f	Pulse frequency of laser
t	Pulse duration
λ	Wavelength of laser
ϕ	Horizontal divergence of the laser
y	Height of laser projection
$nohd$	Maximum allowable safety distance
τ	Width of the optical bandpass filter

Table 3.3: Input parameters and their description of the proposed model.

Chapter 4

Prototype

Using the requirements and the previously presented sensor model, a prototype can be deduced. In the next sections, we first derive the set of sensor parameters which best match the requirements. Additionally, the theorized limitations are presented after which a physical prototype is built. This prototype (dubbed LaserPi) is calibrated in the remainder of this chapter, followed by a final list of parameters and limit estimations based on the calibrated values.

4.1 Design

Before we present the list of selected sensor parameters, we first revisit the requirements in the section. Next, the camera matrix, triangulation parameters, and eye-safety limits are derived. Finally, the theorized, estimated limits are computed.

4.1.1 Requirements and parameter overview

Recalling the Introduction (Chapter 1, the full set of requirements for the prototype are:

1. Solid-state with a cost of $<$ \$30 each, when producing 1000 pieces.
2. Dimension should be within 10x3x4cm.
3. $>$ 2 meter detection range with a resolution $<$ 1%.
4. Planer, horizontal field-of-view of 120 degrees with a resolution of $<$ 1 degree.
5. $>$ 10Hz update rate. That is, 10 times a full 120-degree scan per second.
6. Eye-safe, Class 1 Laser product according to the NEN IEC 60825-1:2014 [24]
7. Fully embedded processing.

In Chapter B, these requirements have been used to derive a set of hardware components, summarised in Table 4.1. Table 4.2 shows an overview of all input-parameters for the model and the parameters which are defined by the hardware selection.

Component	Description
Camera Board	IMX219, mode 5, 1640x922 pixels, 0-60Hz
Processing Unit	Raspberry Pi Zero Wireless
Lens	LS50125, $\sim 133^o$ field-of-view
Laser	25mW, 808nm, 120^o field-of-view
Bandpass filter	808 ± 25nm

Table 4.1: Short list of selected components. For more detailed information on the selection and component properties, see Chapter B.

	Value	Description
$[I_x\ I_y]$	$[1664, 928]$	Imager dimension in pixels
KK	tbd	Camera matrix
$[k_1\ k_2\ k_3]$	tbd	Lens distortion parameters
$[p_1\ p_2]$	tbd	Tangential distortion parameters
s	tbd	Distance between the laser and camera.
α	tbd	Rotation of the camera with respect to s.
β	tbd	Rotation of the laser with respect to s.
ψ	tbd	Rotation of the laser over its central axis.
d	1 mm	Laser diameter
f	tbd	Pulse frequency of laser
t	17.4 ms	Pulse duration
λ	808 nm	Wavelength of laser
ϕ	120^o	Horizontal divergence of the laser
y	4 mm	Height of laser projection
$nohd$	tbd	Maximum allowable safety distance
τ	50 nm	Width of the optical bandpass filter

Table 4.2: Updated list of input parameters, after selecting the used hardware components. Values which are defined as 'tbd' are yet to be selected, derived or estimated.

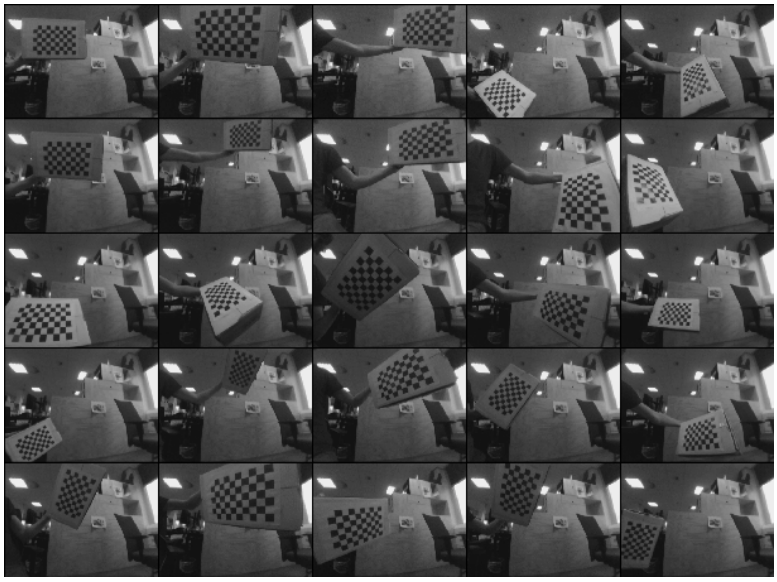


Figure 4.1: Thumbnails of all 25 images used to estimate the Camera Matrix.

4.1.2 Camera matrix estimation

To estimate the Camera Matrix, KK, the Caltech Camera Calibration Toolbox for Matlab [12] is used. For the calibration process, 25 different frames have been captured, depicted in Figure 4.1. These frames show a utilized checkerboard pattern of 7x10 squares with sides of 2.45cm. The estimated camera parameters are shown in Table 4.3. As the toolbox suggests to ignore parameters when their variance is larger than their error, several computations have been made. For each table, the first two columns indicate which parameters are ignored ('0') or used ('1').

The final estimations provide the following camera and distortion values:

$$KK = \begin{bmatrix} 729.586 & 0 & 787.533 \\ 0 & 726.684 & 576.404 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[k_1\ k_2\ k_3] = [0\ -0.07085\ 0.02052]$$

$$[p_1\ p_2] = [0\ 0]$$

$skew$	$k_1k_2k_3p_1p_2$	f_x	f_y	c_x	c_y	$skew$
1	$[1\ 1\ 1\ 1\ 1]^T$	732.6 ± 8.0	729.4 ± 8.0	788.4 ± 6.8	582.4 ± 6.8	0.001 ± 0.003
0	$[1\ 1\ 1\ 1\ 1]^T$	731.7 ± 7.7	728.5 ± 7.6	789.3 ± 6.4	581.7 ± 6.5	0 ± 0
0	$[0\ 1\ 1\ 1\ 1]^T$	731.0 ± 7.0	727.8 ± 6.9	789.5 ± 6.3	581.7 ± 6.5	0 ± 0
0	$[0\ 1\ 1\ 0\ 0]^T$	729.6 ± 6.9	726.7 ± 6.8	787.5 ± 4.9	576.4 ± 4.4	0 ± 0

(a) Estimates for the Camera Matrix. Note that all these values are in pixels. As Mode 4 is used, the values need to be multiplied by $2 * 1.12\mu m$ to get metric values, which yield an estimated focal point of 1.63mm for the last row, in line with the stated focal length in the datasheet of the lens.

$skew$	$k_1k_2k_3p_1p_2$	k_1	k_2	k_3
1	$[1\ 1\ 1\ 1\ 1]^T$	0.004 ± 0.012	-0.076 ± 0.014	0.022 ± 0.005
0	$[1\ 1\ 1\ 1\ 1]^T$	0.002 ± 0.012	-0.075 ± 0.013	0.022 ± 0.005
0	$[0\ 1\ 1\ 1\ 1]^T$	0 ± 0	-0.072 ± 0.003	0.021 ± 0.002
0	$[0\ 1\ 1\ 0\ 0]^T$	0 ± 0	-0.071 ± 0.003	0.021 ± 0.002

(b) Estimates for Radial distortion.

$skew$	$k_1k_2k_3p_1p_2$	p_1	p_2
1	$[1\ 1\ 1\ 1\ 1]^T$	0.002 ± 0.001	0.001 ± 0.001
0	$[1\ 1\ 1\ 1\ 1]^T$	0.002 ± 0.001	0.001 ± 0.001
0	$[0\ 1\ 1\ 1\ 1]^T$	0.002 ± 0.001	0.001 ± 0.001
0	$[0\ 1\ 1\ 0\ 0]^T$	0 ± 0	0 ± 0

(c) Estimates for Tangential distortion.

$skew$	$k_1k_2k_3p_1p_2$	error_x	error_y
1	$[1\ 1\ 1\ 1\ 1]^T$	1.06483	1.13392
0	$[1\ 1\ 1\ 1\ 1]^T$	1.06563	1.13372
0	$[0\ 1\ 1\ 1\ 1]^T$	1.06527	1.13421
0	$[0\ 1\ 1\ 0\ 0]^T$	1.06867	1.13542

(d) Projection error in pixels.

Table 4.3: Calibration results for the LS-50125 lens with mode 4 ($1640x1232$ pixels, 2-binning). The first 2 columns of each table indicate with a boolean which parameters are estimated.

4.1.3 Triangulation deduction

Instead of testing different values for the triangulation parameters (α, β, ψ, s), some initial simplifications can be made:

Effect of β: When the effect of β on q is evaluated, the following can be observed: distance q is computed perpendicular to s, where s is assumed to be parallel to the vertical dimension of the complete sensor. As q is computed based on the reflection of the laser, only objects positioned in the laser-plane are detected. If the laser points downwards ($\beta < 0.5\pi$), objects at a distance larger than $tan(\beta) * s$, have to be located below the sensor. So, when the sensor is placed close to the ground, this property limits the maximum detection distance to $\sim tan(\beta) * s$. As it is likely that larger distances need to be detected, we select $\beta = 0.5\pi$ (or 90 degrees).

Effect of ψ: Ideally, the laser is horizontally aligned with the imager. As such, ψ can be estimated to be 90 degrees.

Effect of s: To analyse the effect of s on the triangulation performance, we need to recall the following equation from Chapter 3.2:

$$q = s * \frac{sin(\beta) * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2 + x_n^2}}{sin(\theta) + y_n * cos(\theta) + \frac{x_n}{tan(\psi)}} \tag{4.1}$$

From this equation, it can be observed that the object distance q (and hence the resolution) is linear dependent on s. Because s is independent of other triangulation parameters, we can assume $s = 1$ for the sensor model, and select an appropriate value after all other parameters have been derived. Moreover, as the sensor is allowed to have a maximum height of 10cm (including the dimensions of the hardware components) the maximum value for s is estimated to be 8cm. Given the requirement that the sensor should be able to observe at least distances of 2 meter, the minimal q-value for $s = 1$ needs to be at least $\frac{200}{8} = 25$ cm.

At this point, all sensor-parameters, except for α have been selected. As such, analysis can be done with different α values. Figure 4.2 shows the minimum and maximum object-distance q, over the range $\alpha \in [50, 120]$ (x-axis, degrees), per imager-column (y-axis, pixels) at which the resolution is $< 1\%$ and $q > 25$cm. Estimating the resolution is done by assuming a sub-pixel accuracy of $1/5^{th}$ of a pixel. That is, we assume that the pixel position of the laser reflection can be estimated as accurately as $1/5^{th}$ of a pixel. Literature shows that values in the region of $[1/2, 1/7.5]$ pixels are possible [29, 33, 19, 35, 17, 38], with some papers even claiming accuracies up to $1/50^{th}$ [9, 26].

In Figure 4.2 a wide variety of q values can be observed. Interestingly, the figure shows that the horizontal center of the frame produces lower q-values when compared to the edges. An effect which is caused by the lens distortion. As low q values require large

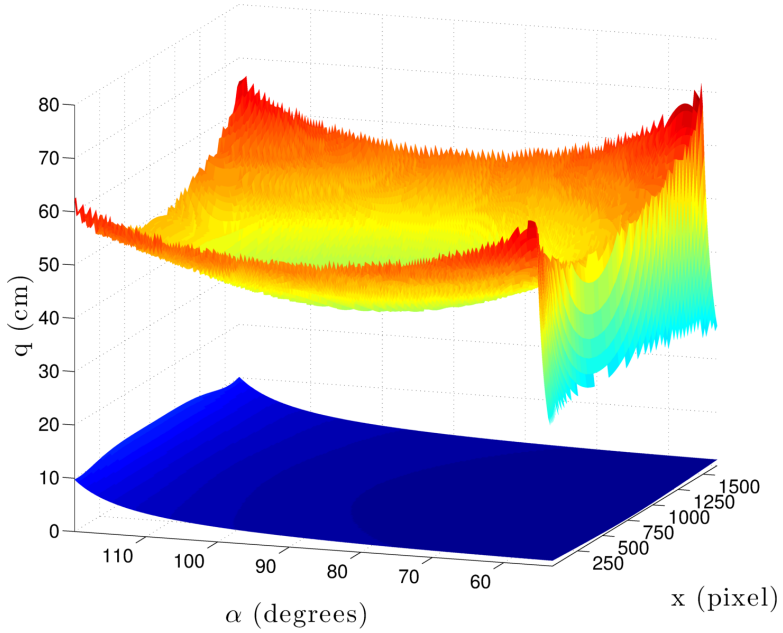


Figure 4.2: Minimum and maximum q-values per α-imager position with the resolution $< 1\%$, $q > 25$ and $s = 1$, assuming a sub-pixel accuracy of 0.2 pixels.

s values to reach the minimal observable distance, it can be concluded from the image that the center of the imager ($x \sim 830$ pixels) is most limiting in terms of resolution. In addition, it can be observed that α-values around 59 or 120 degrees produce the highest q-values. Furthermore, Figure 4.2 shows that the minimal distance of $\alpha \sim 59$ is lower when compared to higher α-values, hence providing the largest and most accurate detection range. More detailed analysis showed that $\alpha = 58.9$ degrees provided the highest q-value (estimated to be 48.9cm) and is therefore selected as input-parameter.

With the selection of $q = 48.9$cm for $s = 1$cm and $\alpha = 58.9$ degrees, the minimal value for s to observe at least 2 meters is $\frac{200}{48.9} \sim 5$cm. In Figure 4.3 the vertical imager position and resolution per q-value are shown for different s-values.

4.1.4 Eye safety and background irradiance

To select the most appropriate s-value, the effect of background irradiance needs to be considered. From Table 4.1 it can be observed that only the pulse frequency and NOHD of the laser systems have to be analyzed. Assuming that a user is not able to get its eyes closer than 1 cm to the laser source and that the laser pulses at the minimum required frequency of 10Hz, the maximum allowable laser power is $30.43mW$, with an NOHD of 0.8cm for the selected $25mW$ laser.

Using a $50nm$ bandpass filter, the background irradiance is estimated to be around $5.1mW/cm^2$. Figure 4.4 shows the laser intensity as a function of the background irra-

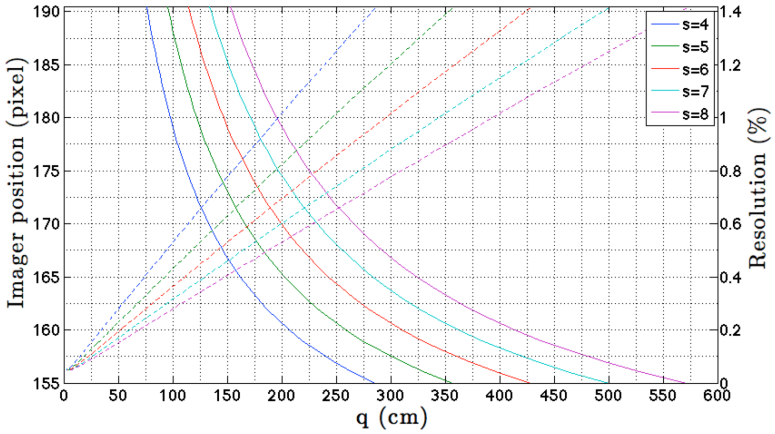


Figure 4.3: Imager position (vertical) and resolution per measured distance of different s-values, with $\alpha = 58.9$ degrees. The dotted line represents the resolution at the stated distances whereas the solid line shows the distance per pixel index. As camera mode 4 is used for testing purposes, while mode 5 will be used in the prototype, the lowest index is 155. For more details see Chapter B.

diance and detection distance. From this figure, it can be derived that at $\sim$ 350cm, the laser reflection is less than 1%. Even if indoor irradiance is assumed, which is typically less then 20% of the outdoor value [36], the laser intensity stays low at $\sim$ 5% of the background irradiance.

4.1.5 Parameter selection and limit estimation

Combining all estimates, s is selected to be $5.5cm$. The complete list of input parameters is shown in Table 4.4. Using these values, the estimated maximum detection distance is $\sim$ 375cm. Depending on the sub-pixel accuracy, different resolutions can be achieved as shown in Table 4.5. To meet the requirement for a maximum error of 1% at 2 meters, the position of the laser reflection has to be detected within 0.5 pixels.

Using the selected parameters a prototype is designed as shown in Figure 4.5.

4.2 Calibration

Physical systems commonly need a calibration step in order to correct for misalignment errors and other irregularities. This section presents the camera calibration setup of the prototype, followed by the estimation of the laser plane. The latter is an extension build on top of the earlier mentioned Caltech Toolbox (Chapter 4.1.2), allowing calibration of both camera and triangulation with the same set of frames. This chapter is concluded with an updated analysis of the expected limitations of the prototype, using the calibration parameters.

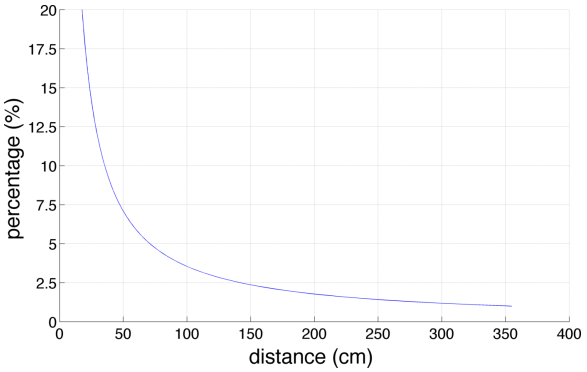


Figure 4.4: Reflection intensity of the 25mW, 120^o line laser with respect to the background irradiance when using a 50nm bandpass filter. At 100cm, the laser reflection is only $\sim 3.25\%$ of the background irradiance. Distances larger then $\sim$ 350cm are unlikely to be detected as the reflection drops below 1% of the background irradiance.

	Value	Description
$[I_x\ I_y]$	$[1664, 928]$	Imager dimension in pixels
KK	$\begin{bmatrix} 729.586 & 0 & 787.533 \\ 0 & 726.684 & 576.404 \\ 0 & 0 & 1 \end{bmatrix}$	Camera matrix
$[k_1\ k_2\ k_3]$	$[0\ -0.07085\ 0.02052]$	Lens distortion parameters
$[p_1\ p_2]$	$[0\ 0]$	Tangential distortion parameters
s	5.5cm	Distance between the laser and camera.
α	58.9^o	Rotation of the camera with respect to s.
β	90^o	Rotation of the laser with respect to s.
ψ	90^o	Rotation of the laser over its central axis.
d	1 mm	Laser diameter
f	10 Hz	Pulse frequency of laser
t	17.4 ms	Pulse duration
λ	808 nm	Wavelength of laser
ϕ	120^o	Horizontal divergence of the laser
y	4 mm	Height of laser projection
$nohd$	0.8 cm	Maximum allowable safety distance
τ	50 nm	Width of the optical bandpass filter

Table 4.4: Complete list of selected input parameters of the sensor model.

sub-pixel accuracy	$min(q)$	200cm	$max(q)$
0.1	0.03%	0.39%	0.75%
0.2	0.05%	0.78%	1.49%
0.3	0.08%	1.17%	2.22%
0.4	0.10%	1.55%	2.94%
0.5	0.13%	1.91%	3.65%
0.6	0.16%	2.29%	4.35%
0.7	0.18%	2.67%	5.04%
0.8	0.21%	3.05%	5.72%
0.9	0.24%	3.41%	6.39%
1.0	0.26%	3.79%	7.05%

Table 4.5: Estimated resolution with with different sub-pixel accuracy laser localisation values. The minimum and maximum distance which can be detected are limited to the imager dimensions, and therefore do not change with the detection accuracy. These limits are estimated to be 2.4cm and 375cm respectively.

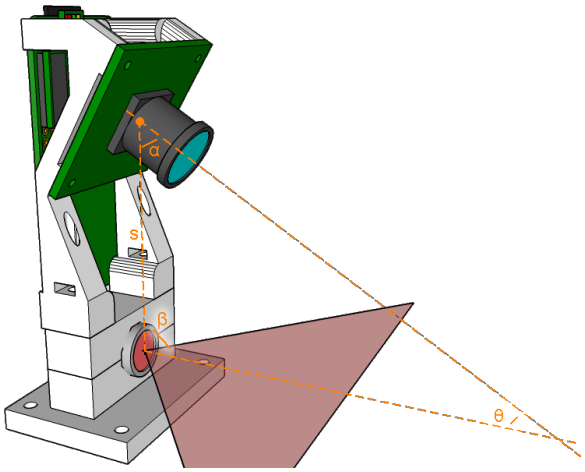


Figure 4.5: 3D rendering of the prototyped setup. The optical centre is estimated to be located at 2.4mm, measured from the back of the camera board. The laser is placed such that its central axis is at a vertical offset of 55mm from this point.

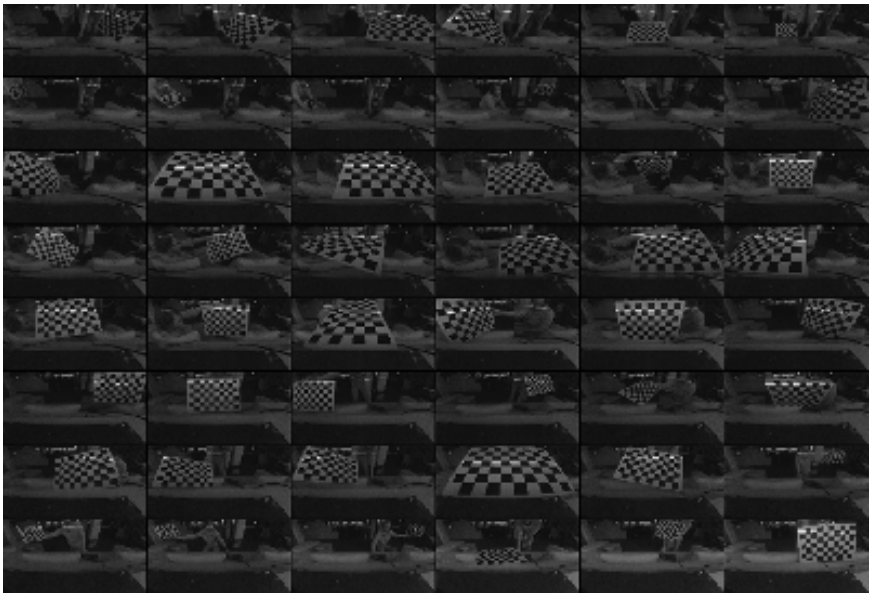


Figure 4.6: Overview of the calibration images, taken by the prototype. Due to the usage of a bandpass filter, the images are darker when compared to the calibration images presented in the previous chapter.

4.2.1 Camera

The camera calibration is done following the same steps as described in Chapter 4.1.2, except, in order to get better estimates, more images have been captured and a larger checkerboard pattern has been used. In total 48 frames have been used, depicted in Figure 4.6. The checkerboard has been resized to squares with sides of 4.14cm, still consisting of a pattern of 7x10 squares. Table 4.6 shows the estimated camera parameters. When comparing these results with Table 4.3, it can be noted that the accuracy has improved.

4.2.2 Triangulation

To compute the camera parameters, the Caltech Toolbox needs to compute and estimate a different projection for each image. This projection provides the information to compute the backward projection of a checkerboard-pixel to a 3D point, projected in the camera reference frame. When a checkerboard intersects with the laser-plane, a laser reflection is observed. Because a pixel coordinate of the laser reflection on the checkerboard is essentially a checkerboard-pixel, we can transform this reflection into a 3D point too. By collecting several 3D points of the laser reflection from different images we essentially create a set of samples of the laser plane. By fitting a plane through the set of samples, the laser-plane parameters, (a_c, b_c, c_c), can be estimated. As this plane is computed from the actual setup, it implicitly contains all corrections for misalignment.

$skew$	$k_1k_2k_3p_1p_2$	f_x	f_y	c_x	c_y	$skew$
1	$[1\ 1\ 1\ 1\ 1]^T$	592.2 ± 1.6	591.0 ± 1.6	802.3 ± 1.2	413.2 ± 1.5	$\sim 0 \pm \sim 0$
0	$[1\ 1\ 1\ 1\ 1]^T$	592.1 ± 1.6	591.0 ± 1.6	802.4 ± 1.1	413.2 ± 1.5	0 ± 0

(a) Estimates for the Camera Matrix. As Mode 5 is used, the values need to be multiplied by $2 * 1.12\mu$ to get metric values. This results in a focal point estimation of 1.3mm, differing slightly from the datasheet which states 1.6mm. This error might be introduced due too the use of a different lens and cameraboard, the addition of a bandpass filter or difference in focus distance.

$skew$	$k_1k_2k_3p_1p_2$	k_1	k_2	k_3
1	$[1\ 1\ 1\ 1\ 1]^T$	-0.021 ± 0.002	-0.039 ± 0.002	$0.009 \pm \sim 0$
0	$[1\ 1\ 1\ 1\ 1]^T$	-0.021 ± 0.002	-0.039 ± 0.002	$0.009 \pm \sim 0$

(b) Estimates for Radial distortion.

$skew$	$k_1k_2k_3p_1p_2$	p_1	p_2
1	$[1\ 1\ 1\ 1\ 1]^T$	$-0.003 \pm \sim 0$	$\sim 0 \pm \sim 0$
0	$[1\ 1\ 1\ 1\ 1]^T$	$-0.003 \pm \sim 0$	$\sim 0 \pm \sim 0$

(c) Estimates for Tangential distortion.

$skew$	$k_1k_2k_3p_1p_2$	error_x	error_y
1	$[1\ 1\ 1\ 1\ 1]^T$	0.56940	0.55007
0	$[1\ 1\ 1\ 1\ 1]^T$	0.56968	0.54982

(d) Projection error in pixels.

Table 4.6: Camera calibration results of the prototype. The first 2 columns of each table indicate with a boolean which parameters are estimated.

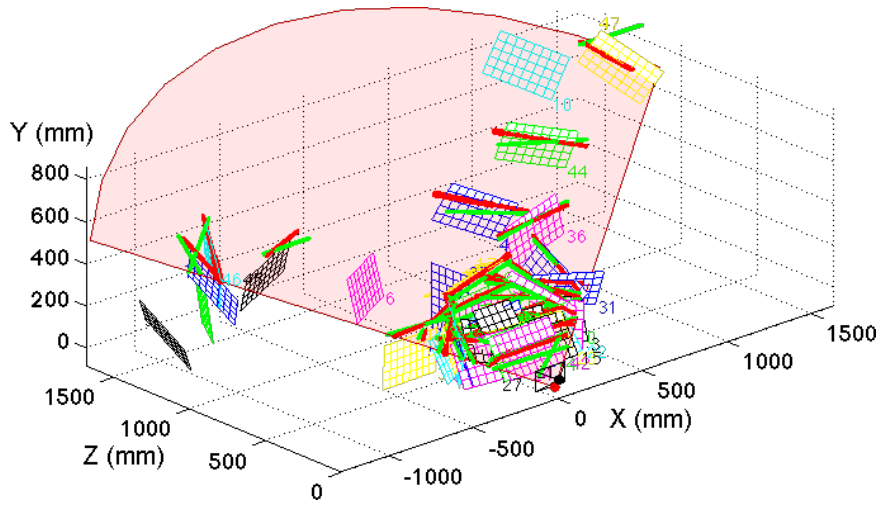


Figure 4.7: Projection of the estimated laser plane (red). The green dots are the projected position of the laser reflection, computed by the laser plane. The red dots are the same reflection, but projected by the per-image based projection of the Caltech Toolbox. The estimated positions of the checkerboards are displayed in a wide variety of colors. All points are displayed in the camera reference frame

From the triangulation parameters provided in Table 4.4, the (ideal) laser-plane parameters can be deduced:

$$(a_i, b_i, c_i) = (0, 1.6, 106.5)$$

Using a least-square fit, the calibrated plane is estimated to be:

$$(a_c, b_c, c_c) = (0.03068, -1.62396, 94.65943)$$

The least-square error of the estimated laser-plane is 6.26mm, translated into a displacement error of ± 2.8mm for all calibration frames. Figure 4.7 shows the estimated 3D positions of the checkerboard in the camera reference frame. Additionally, it shows the projected 3D positions of the laser reflection, estimated by using the per-image based projection of the Caltech Toolbox and the derived laser-plane parameters.

While the least-square error is relatively small there are large projection errors observable between the Caltech and the laser-plane projections. Ideally, both should align, but as the pixels for the laser reflection are manually selected, some local errors may be introduced. Resolving these errors is done by iteratively computing the difference between both projections and adjusting the selected pixel position vertically, with 10 percent of the error. In each iteration, all images are processed after which a new laser-plane is estimated. Table 4.7 shows the mean-square error of the difference of both projections for each iteration.

iteration	mean(error^2)
1	518.84081
2	206.57550
3	93.50105
4	46.17197
5	24.27914
6	13.39586
7	7.68133
8	4.54695
9	2.76469
10	1.71999
11	1.09152
12	0.70488
13	0.46232
14	0.30752
15	0.20721

Table 4.7: Mean-square error between the per-image based Caltech projection and the globally fitted plane. Each iteration the pixel position of the projected point is linearly corrected with the difference between the two projections.

After optimisation, the laser plane is estimated to be:

$$(a_c, b_c, c_c) = (0.03099, -1.62720, 94.05262)$$

with a least-square error of only 00.01mm.

When comparing the calibrated plane parameters and the parameters derived directly from the proposed model, a displacement over x (since $a_c \neq 0$) and a significant estimation error of the optical center position of the lens can be observed ($c_i \neq c_c$). The former suggests that the laser has a roll of 1.49 degrees, whereas the latter shows that the optical center is not located at 2.4mm, but at 13.6mm, measured from the back of the Camera Board. A complete comparison of the triangulation parameters is shown in Table 4.8.

4.2.3 Limit estimation

Since several input-parameters have changed due to the inclusion of misalignment error, the originally predicted limitations of the sensor need to be updated. In Table 4.9 an updated overview of all input parameters is given. Only the camera and triangulation parameters are updated, as these are affected by the calibration procedure.

Using these values, the minimum value of q is estimated to be 1.4cm, while the maximum goes to infinity. However, due too the background irradiance only distances up to ~ 3.50 might be observable. In Table 4.10 the estimated resolution for different sub-pixel accuracies is shown.

parameter	original	calibrated	optimised
θ	31.10^o	31.62^o	31.57^o
ψ	90.00^o	91.63^o	91.51^o
α	58.90^o	58.38^o	58.43^o
s	5.50 mm	49.63 mm	49.24 mm

Table 4.8: Comparison of the estimated triangulation parameters, defined originally, after calibration and after optimisation of the laser plane. For all values it is assumed that $\beta = 90$ degrees and that both camera and laser are perfectly aligned with no unmodelled errors.

	Value	Description
$[I_x\ I_y]$	$[1644, 928]$	Imager dimension in pixels
KK	$\begin{bmatrix} 592.139 & 0 & 802.362 \\ 0 & 590.933 & 413.244 \\ 0 & 0 & 1 \end{bmatrix}$	Camera matrix
$[k_1\ k_2\ k_3]$	$[-0.02077, -0.03930, 0.00889]$	Lens distortion parameters
$[p_1\ p_2]$	$[-0.00259, 0.00025]$	Tangential distortion parameters
s	4.92cm	Distance between the laser and camera.
α	58.43^o	Rotation of the camera with respect to s.
β	90.00^o	Rotation of the laser with respect to s.
ψ	91.51^o	Rotation of the laser over its central axis.
d	1 mm	Laser diameter
f	10 Hz	Pulse frequency of laser
t	17.4 ms	Pulse duration
λ	808 nm	Wavelength of laser
ϕ	120^o	Horizontal divergence of the laser
y	4 mm	Height of laser projection
$nohd$	0.8 cm	Maximum allowable safety distance
τ	50 nm	Width of the optical bandpass filter

Table 4.9: List of input parameters for the sensor model, after calibrating the prototype. It should be noted that the computed triangulation parameters assume the value of β and that all possible misalignments are modelled in the sensor model.

sub-pixel accuracy	$min(q)$	200cm	$250cm$	$350cm$
0.1	0.04%	0.53%	0.65%	0.90%
0.2	0.08%	1.05%	1.30%	1.79%
0.3	0.12%	1.57%	1.92%	2.67%
0.4	0.16%	2.09%	2.56%	3.54%
0.5	0.21%	2.58%	3.21%	4.41%
0.6	0.25%	3.10%	3.81%	5.22%
0.7	0.29%	3.58%	4.41%	6.03%
0.8	0.33%	4.08%	5.01%	6.88%
0.9	0.37%	4.55%	5.61%	7.65%
1.0	0.41%	5.05%	6.19%	8.45%

Table 4.10: Estimated resolution for different sub-pixel accuracy localisation values for the calibrated prototype. The minimum and maximum distance which can be detected are limited to the imager dimensions and background irradiance and do not change with the sub-pixel accuracy. These limits are estimated to be 1.4cm and 350cm respectively.

Chapter 5

Laser-Camera Synchronisation

The proposed laser distance sensor operates by capturing the reflection of a laser source. Naturally, this needs an excited laser while the light-sensitive part of the camera is exposed. This chapter introduces and explains different methods of synchronization, to synchronize the laser and camera exposure. After an introduction tests and limit estimation of the selected mechanism is performed.

5.1 Synchronisation

Several synchronization mechanisms exist, ranging from hardware- to software- controlled, or a combination of those. By including the limits of the Raspberry Pi processing board and the selected IMX219 imager, this section shows potential techniques.

5.1.1 Continuous activation

A naive approach to synchronize the laser is to excite it continuously: regardless of the exposure time or capture frequency of the camera, the laser will always be active when the camera generates a frame. However, as stated in Chapter 3.3, the maximum allowable energy output for a laser to meet its class restriction, is inversely related to the duration it is active. Hence a continuous activated laser greatly limits the maximum allowable power output of the laser which has a negative influence on the distance at which a laser reflection can be detected.

5.1.2 PWM controlled

When the framerate of the camera is known, a slightly more intelligent solution can be used, as described in [19]. As Gao et. al. did not have access to low-level camera control, they opted to drive the laser with a PWM signal at half the frequency of camera, with a 50% duty cycle. As a result, the laser is visible in every other frame. This approach allows a higher energy output when compared to the naive approach and introduces an extra dimension to detect the laser: the laser can be tracked by searching for pulsing pixels.

5.1.3 Low-level control

If low-level control is available, such as the 'vsync' signal [33] or 'strobe' line [4], even better timing can be achieved. By synchronizing the exposure time at a hardware level, the active period of the laser can be set close to the exposure time. Measuring or estimating the delay between frame exposure and the 'vsync' signal allows the build of a delayed controller which activates the laser upon the next frame exposure. This mechanism requires a fixed framerate. Utilising the 'strobe' properties of an imager provides even more control. CMOS imagers use the 'strobe' signal to activate a flash. Precise timings are tuned with registers on the camera chip and are to a certain extent framerate independent.

While the datasheet [4] of the selected IMX219 imager contains some information on registers which affect the 'strobe' signal, actual functioning register settings are unknown. Determining suitable register settings commonly requires a lot of time/effort and close communication with the chip supplier. Since both are not available in this thesis, this approach is out of scope. For utilizing the 'vsync' signal yields the same: as we are using off-the-shelf components, using a hardware signal would require several modifications on the PCB.

5.1.4 GPU controlled

Documentation of a Python library for the IMX219 shows that the GPU of the Raspberry Pi is able to control a LED flash, synchronized with the exposure of the imager [27]. As the GPU runs a real-time kernel, this approach is the closest in term of direct hardware control as we can get with the current setup.

Tests with a GPU-controlled flash have been performed for both 'capture' and 'video' mode. The findings can be summarised as follows:

- capture-mode: the minimum delay between a CPU signal and the callback is around 0.48s. As several CPU-calls cannot be pipelined, the maximum achievable framerate is 2Hz.
- video-mode: in this mode, there is no support for a GPU controlled flash.

The requirements state that the minimum update frequency should be at least 10Hz. Therefore, the GPU controlled flash is not an appropriate approach.

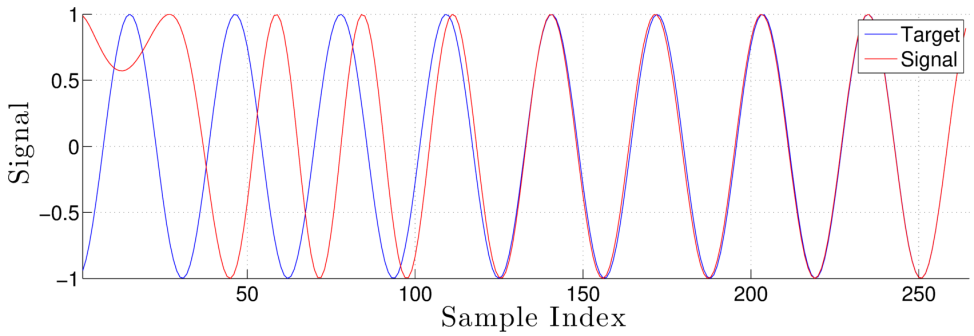


Figure 5.1: Example of the output of a phase lock loop [18]

5.1.5 Software Phase Locked Loop

The GPU provides every frame with a presentation timestamp (PTS) [2, 27]: a PTS is a timestamp in the GPU domain at which the GPU received the first line of a frame from the imager. With the GPU running an RTOS and the imager set at a fixed framerate, this timestamp can be used to estimate the time at which the next frame is generated and hence it can be used to synchronize the laser.

While computations on the CPU might provide estimations for synchronization times, due too the nature of not being an RTOS, the CPU is unable to directly control and synchronize a laser stably. Documentation of the BCM2835 [1] shows that the SoC is able to produce a hardware PWM signal. Such a PWM signal allows us to specify a constant rate and pulse duration at which a laser is active, without jitter or overhead of an operating system or any other software control. Having both the camera and laser active at an equal and constant (hardware) frequency, we only need to synchronize the periods, solvable by utilizing a Phase Lock Loop (PLL) [23].

A PLL consists of a fixed frequency signal (PWM/laser) to which an adjustable signal (imager) is matched. By computing the offset between these signals, a controller adjusts the frequency of the adjustable signal until the signals are in phase. In Figure 5.1 an example is shown of two sinusoids.

For our prototype, this controlling update is done each time a frame is received. Details on how this control is build is explained in the next section.

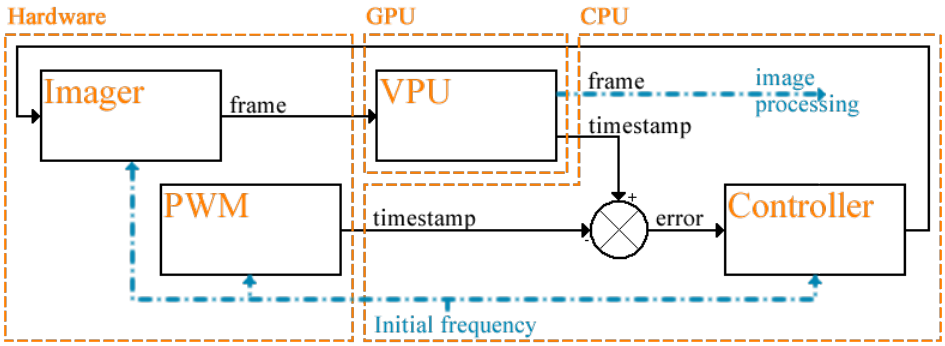


Figure 5.2: Controller setup with annotation which blocks are operated in what domain: hardware (no software interrupts), GPU (real time) or CPU. The blue lines are added to provide a full overview of all processes. They might initialise, but do not influence the control loop.

5.2 Setup

5.2.1 Overview

Several components need to cooperate in order to synchronise the laser with the imager. In Figure 5.2 an overview is given of how these components are connected and in what domain they operate.

With the selected hardware there are three identifiable domains: hardware, GPU and CPU controlled:

- Hardware: blocks which operate without interrupts. Once initialized, they keep running at a set frequency or speed.
  - Imager, producing an image/frame.
  - PWM, controlling the laser.
- GPU: blocks which can be interrupted by external processes. But as the GPU runs an RTOS, timing is accurate.
  - VPU (see Figure B.1), adding a timestamp to a frame.
- CPU: blocks which are interrupted by external processes.
  - Controller, triggered when a new frame is available. Computes the phase difference (error) between the presentation time of the frame and the timestamp of nearest PWM pulse. It computes a new frequency for the Imager based on this error.

5.2.2 Time-stamping and error estimation

While the GPU adds a GPU-domain timestamp to each timeframe, the PWM signal does not has such a signal. The hardware PWM can be initialized with a certain duty cycle and frequency, but no interrupts are triggered when the laser is activated. Therefore, a PWM timestamp in the GPU-clock domain needs to be computed based on the (CPU) timestamp at which the PWM signal is started. In Algorithm 1 the used process is shown.

Algorithm 1 Starttime estimation of PWM in the GPU domain

```
1:  function INIT_PWM(interval_max)
2:      repeat
3:          T_1 ← TIME(cpu)
4:          START_PWM()
5:          T_gpu ← TIME(gpu)
6:          T_2 ← TIME(cpu)
7:          interval ← T_2 − T_1
8:      until interval < interval_max
9:      T_pwm ← T_gpu − interval
10:     return ( T_pwm, interval)
```

As the initialization code for the PWM runs on the CPU, the operating system might interrupt the function calls which set and start the PWM mechanism. Therefore we cannot determine the exact timestamp at which the PWM signal is started. In Algorithm 1 an approximation is computed by determining the interval (line 3 and 6) in which the PWM signal is started. If the interval is larger then a predetermined maximum (line 8), the PWM system is restarted. This process repeats until the estimation is within the predetermined bound.

In the same interval, the current time of the GPU is clock is also requested (line 5). Using the GPU time and the CPU interval the start-time of the PWM signal can be estimated to be within $[\mathcal{T}_{pwm}, \mathcal{T}_{pwm}+\text{interval})\mu s$ in the GPU domain.

By combining this estimation, the presentation time of the frame and the frequency of the PWM signal, the smallest error between a frame and pulse can be computed.

First we need to estimate the number of pulses after startup, given a PWM-timestamp, $\mathcal{T}_{pwm}$:

$$k = \left\lfloor \frac{\mathcal{T}_{pts} - \mathcal{T}_{pwm}}{p} \right\rfloor \tag{5.1}$$

Where k represents the number of pulses, $\mathcal{T}_{pts}$ the presentation time of the frame, $\mathcal{T}_{pwm}$ the start time of the laser and p the period of the frame capturing duration, all expressed in microseconds.

Using the number of pulses the timestamp of the last pulse can be computed:

$$\mathcal{T}_{last,pwm} = \mathcal{T}_{pwm} + k * p \tag{5.2}$$

Allowing the error, e (in microseconds), to be estimated as:

$$e = \mathcal{T}_{pts} - \mathcal{T}_{last,pwm} + \delta - \frac{interval}{2} \tag{5.3}$$

Where δ is a user defined offset (allowing the inclusion of additional fixed timing delays) and $\frac{interval}{2}$ ensures that we are synchronising the laser with the centre of the interval.

When the error is expressed as a percentage of frame-period ($e_{\%} = \frac{e}{p}$), we can distinguish between a frame being received too early or too late: if $e_{\%} > 50\%$, the received frame is closer to the next laser pulse and is therefore produced $(100\% - e_{\%}) * p$ μs too early. By including this factor, the error is expressed in the range of $(-\frac{1}{2}p, \frac{1}{2}p)$.

5.3 Experiments and results

After estimating the error, this section shows the experiments and results of the accuracy of the synchronization. First, an analysis is done on the boundaries of the PWM-interval, followed by an analysis of the control system to reduce the synchronization error.

5.3.1 PWM interval

Empirical analysis showed that $interval\_max$ could be set as small as 300 μs. Source code analysis additionally shows that this includes a delay of 111 μs before the PWM signal is started, hence the accuracy can be increased to $189\mu s$. In Table 5.1 test results are displayed, showing an average interval of 184.1 ± 3.92 μs, reached in 5.85 ± 2.70 iterations. A larger test set (N=550) resulted in an interval of 185.0 ± 2.79 μs.

5.3.2 Tuning

To reduce the synchronization error, a PID controller [25, 20] is used. Such a controller consists of three tuning parameters, which in short can be described as:

- P: proportional gain; corrects the signal with a value proportional to the error. A large P-value means that the controlled value is more sensitive to the direct error.
- I: integrator; corrects the offset between the controlled value and a target value.
- D: derivative; decreases (damps) the oscillation of the controlled signal.

Before each of these parameters is determined the (step)response of the imager is recorded by toggling the framerate between 30Hz and 32Hz. By logging the presentation timestamp of the received frames, the actual framerate of the imager is computed. The resulting response is shown in Figure 5.3.

As a request to change the framerate is started in the CPU domain and needs to travel through the GPU to finally be processed at the imager, a (varying) delay in the response can be observed. Additionally, as the framerate is a digital parameter, no overshoot, no setup time and a small offset can be observed, which increases the complexity of algebraic

#	interval (μs)	iterations
1	185	3
2	186	6
3	182	12
4	187	4
5	187	3
6	181	3
7	180	3
8	189	4
9	184	5
10	185	4
11	189	5
12	181	9
13	177	3
14	189	6
15	179	7
16	184	5
17	188	9
18	184	8
19	188	10
20	177	8

Table 5.1: Data from several runs of determining the interval in which the PWM is started. Average interval is 184.1 ± 3.92 μs, reached in 5.85 ± 2.70 iterations.

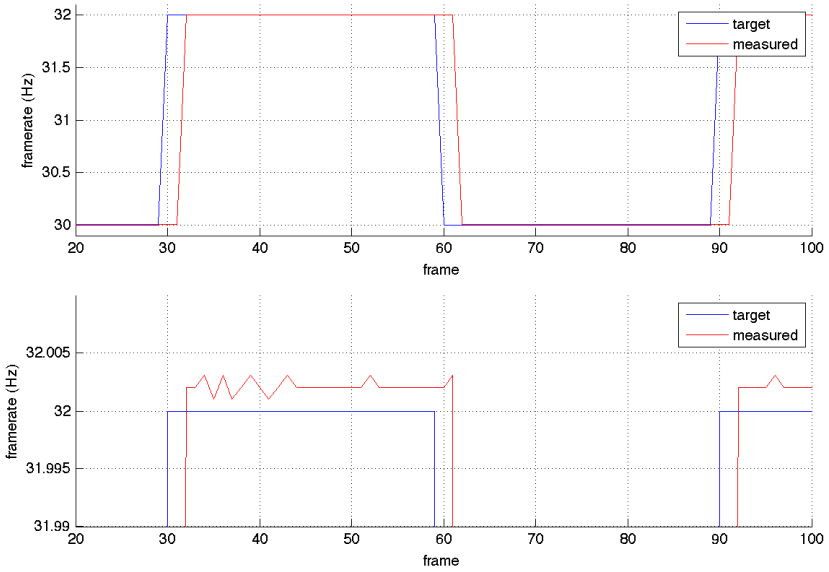


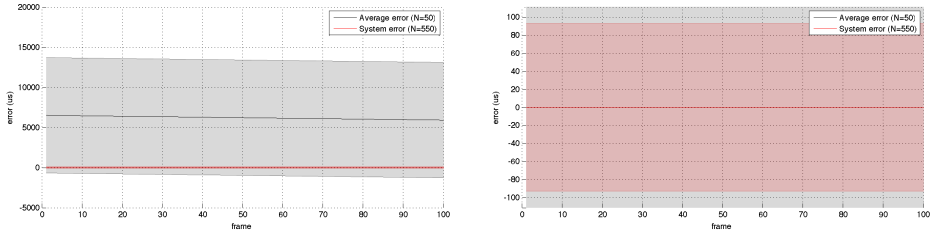
Figure 5.3: Imager response when switching back and forth between a 30Hz and 32Hz framerate, using the imager in mode 5. Top graph shows the full response while the bottom graph shows a more detailed view

tuning of the controller [45, 15]. When factoring in the system error (the estimation of *interval*), manual tuning is assumed to provide sufficient synchronization accuracy.

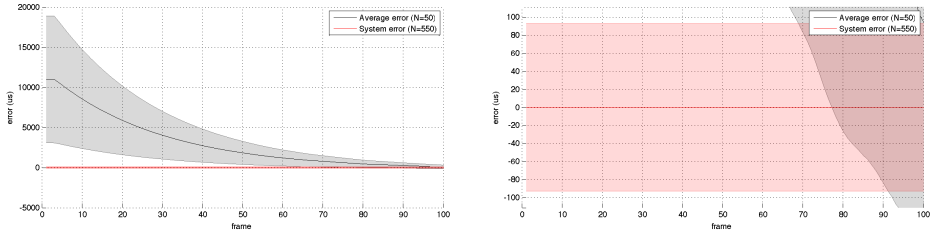
Tuning the controller is done by varying the proportional gain while setting both the integration and derivation to zero. The gain has been tested with values in the range $P \in [0, 9]$ with a step size of 1. As the initial error between the laser and the camera varies with the startup time, each P-value has been tested for 50 startups, for which the first five seconds of the response is recorded. The average response per gain is shown in Figure 5.4 and Figure 5.5.

For a proportional gain, $P > 4$, the synchronization error is within the system error within 30 frames. As the oscillation stays below the system error, additional tuning of the integration and derivation parameter is not needed. From Figure 5.4 and Figure 5.5 it can also be observed that a larger gain results in a smaller error, but it also requires more frames to become stable. Therefore a gain of $P = 7$ is selected: the data suggests that the synchronization error is smaller than the system error within ~ 22 frames, while oscillating a minimum of 10 μs within the system error.

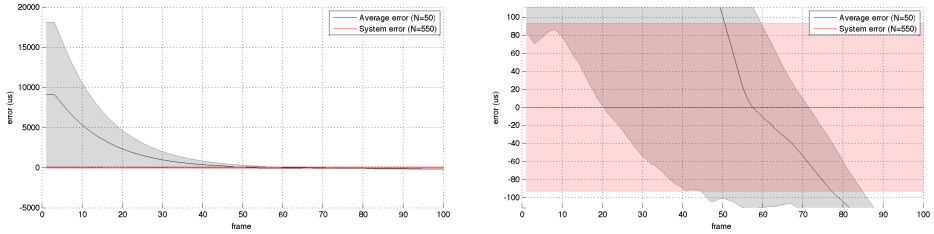
With the correction factor being updated every frame update and being linear dependent on the error and hence the frequency, the gain of $P = 7$ (measured at 30Hz) can be computed as a frequency-dependent gain of $\frac{7}{30} \sim 0.233 * f$.



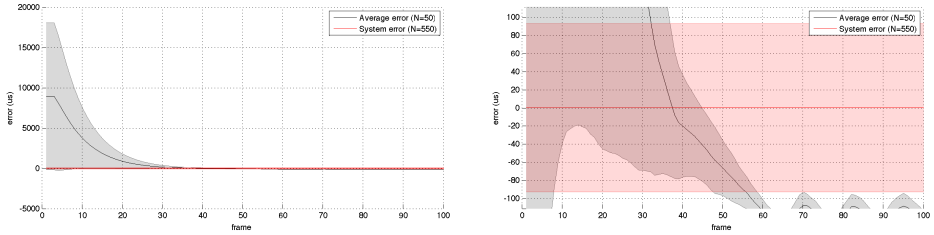
(5.4a) P=0, I=0, D=0



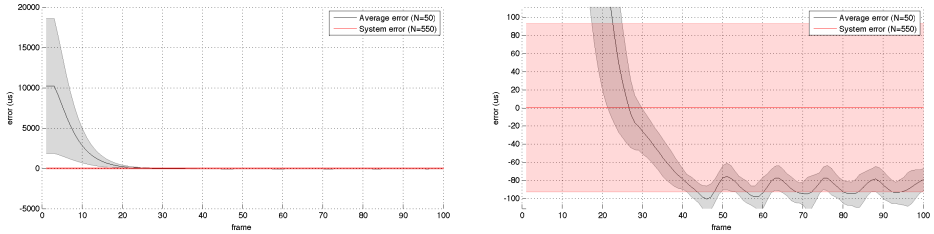
(5.4b) P=1, I=0, D=0



(5.4c) P=2, I=0, D=0

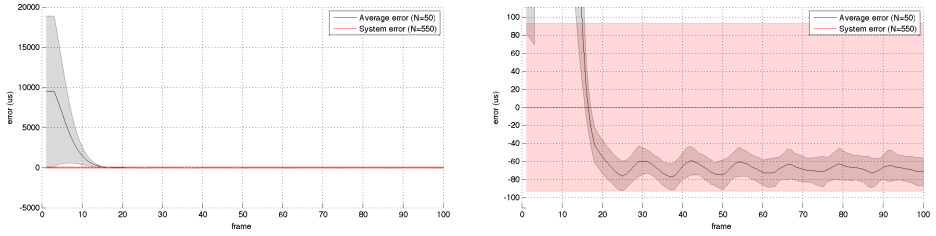


(5.4d) P=3, I=0, D=0

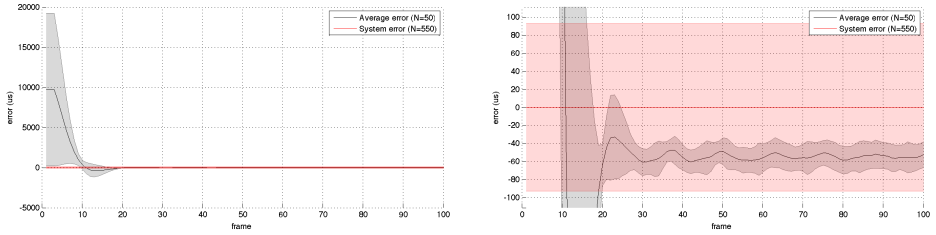


(5.4e) P=4, I=0, D=0

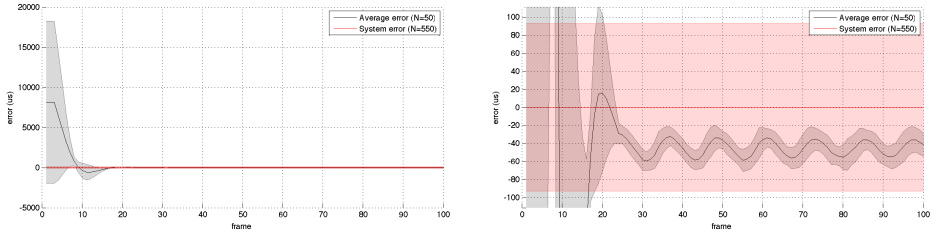
Figure 5.4: Average synchronisation error per gain ($P \in [0, 4]$) of the first 100 frames, for 50 system start-ups. The grey area represents the standard deviation and the red area the system error (*interval*). Left: full view of error. Right: detail, zoomed in on the system error.



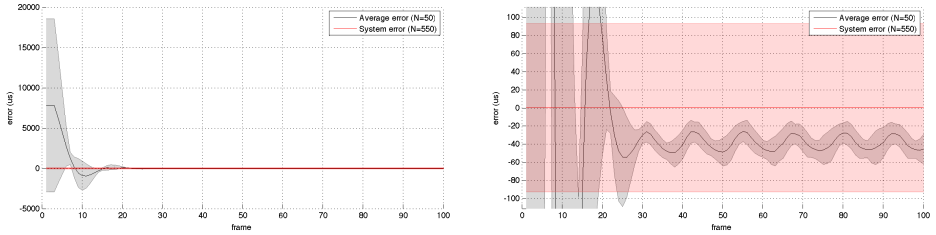
(5.5a) P=5, I=0, D=0



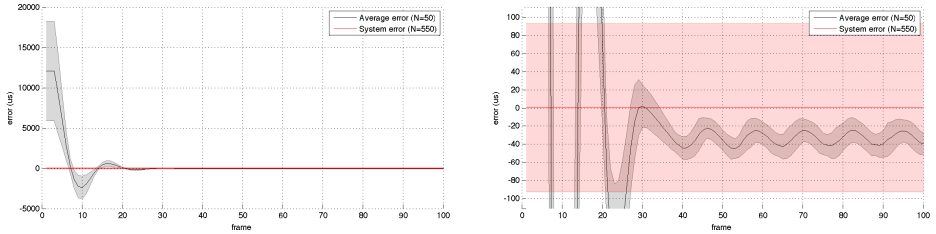
(5.5b) P=6, I=0, D=0



(5.5c) P=7, I=0, D=0



(5.5d) P=8, I=0, D=0



(5.5e) P=9, I=0, D=0

Figure 5.5: Average synchronisation error per gain ($P \in [5, 9]$) of the first 100 frames, for 50 system start-ups. The grey area represents the standard deviation and the red area the system error (*interval*). Left: full view of error. Right: detail, zoomed in on the system error.

Chapter 6

Distance Estimation and Performance

After modeling, calibrating and synchronizing the LaserPi, this chapter presents the developed software pipeline, image filters, corrections and the final distance measurements with their (in)accuracies.

6.1 Image processing

Several processing steps are performed before a final estimation of the laser reflection is given. In the next sections, we first introduce the full processing pipeline followed by a more detailed explanation of the image processing step.

6.1.1 Pipeline

The prototype is built on top of a Raspberry Pi Zero (see Chapter B). At the heart of this system is a BCM2835 SoC [1] containing a 32 bit, single core ARM1176JZFS processor running at 1GHz and a VideoCore IV GPU running at 250MHz, capable of 24 GFLOPS.

When looking at the requirements, it can be noted that we want an update frequency of at least 10Hz. If all processing would be done on the ARM, this would mean that each frame should be processed within 100.000.000 clock pulses. The previous chapters show that the IMX219 will be used in mode 5, producing 1664x928 pixels per frame. Basic image processing operations such a blurring with a 5x5 window or pixel comparisons might require a total of at least 30 operations/readouts per pixel, resulting in a CPU utilization of 45 percent. Since this does not include any flow control, memory calls, context switching or other (inter-)process controls, it strongly indicates that a pure-CPU solution might not be sufficient. As such, a pipeline has been constructed, making use of both the CPU and GPU. An overview is shown in Figure 6.1.

The designed pipeline is controlled by the LaserPi-process, running on the ARM. It initiates the imager and Image Sensor Pipeline (ISP) via the MMAL interface [2]. After

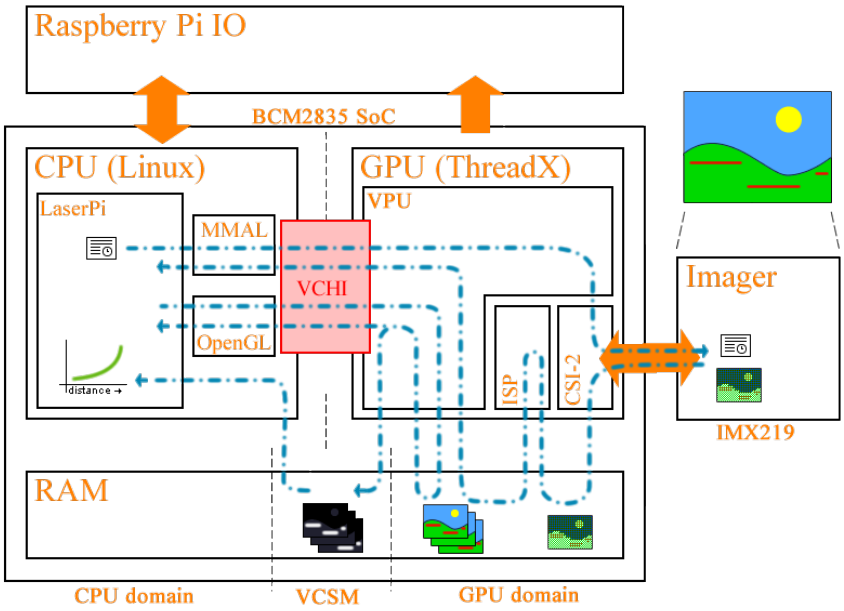


Figure 6.1: Schematic overview of the full processing pipeline. All CPU-GPU communication is handled by the VideoCore Host Interface (VCHI).

initialization, the imager produces raw Bayer-frames at a fixed interval. These frames are received and timestamped by the GPU, which pushes the raw image through the ISP, producing YUV formatted images. The result is stored in a GPU-located image-pool, after which the CPU is notified via an MMAL callback. This callback initiates the OpenGL processing stage on the GPU, which retrieves the ISP-processed image from the pool, applies several filter operations, and stores the result in a second pool in the CPU-GPU shared memory region (VCSM). A second notification is sent to the CPU, which then retrieves the filtered frame from the VCSM, performs a laser-detection step and computes the 3D coordinates of the laser reflection. By utilizing the CPU-GPU shared memory, image buffers do not need to be copied between the GPU and CPU. As such, both the ARM and OpenGL can perform operations on the same memory structure, making more efficient processing possible.

In addition to activating the OpenGL processing, the MMAL callback also updates the framerate of the imager and synchronization of the laser as explained in Chapter 5.

6.1.2 Filtering

The image filtering follows the same approach as Gao et. al [19]. In their setup, they pulse the laser on alternate frames and track the per-pixel luminosity transition. When a pixel has a minimum count of four transitions it is said to be part of the reflection. The resulting binary image is then blurred, interpolated and finally used to do laser localization and distance estimations.

Our approach slightly deviates and is done in the following sequence:

1. After receiving a callback from the MMAL interface, only the luminance channel (Y) of the captured image is pushed to an OpenGL structure. The other channels, U and V, are discarded.

2. We then compute the absolute difference between two consecutive frames. To enhance the detectability of the laser at longer distances this step additionally multiplies the top 20 percent of the filtered image with a linear gain, ranging from 1 to 10.

3. Next, the image is blurred with a 5x5 normalised window, where the weight, w, for each element in the window is represented by its inverse distance:

$$w = 1/\sqrt{dx^2 + dy^2 + 1} \tag{6.1}$$

   with dx and dy representing the relative pixel position with respect to the centre of the window. This step acts as a low-pass filter, reducing the effect of random noise. For more reduction, blurred pixels with a value below 0.1 are set to 0.

4. In contrast to Gao et. al [19], we use a blend instead of a transition tracker. The blend merges the newly filtered image and a running average, resulting in a new average, by performing a pixel blend according to

$$b = a * (1 - blend) + b * blend \tag{6.2}$$

   where $blend$ is a blend factor of 0.75 and a and b the new and running average pixel luminosity respectively.

5. The blend is then analyzed in the CPU domain. On the CPU, we first retrieve the highest pixel index per image-column for the pixel which has the highest luminance value. As pixels near each other are likely to have similar values, this process is sped up by skipping every other column and row.

6. Computing the vertical center of the laser reflection is done by finding for each column the boundaries of the uninterrupted blob, holding at least 70 percent of the value of the previously found maximum, where the maximum has at least a value of 10 (out of 255). The weighted average of this blob is then computed and assumed to represent the center of the laser reflection. When the total weight of the blob is below 25, the blob is discarded as noise.

7. Finally, for each column, the estimated laser-pixel is transformed in a distance estimation, via the backward projection of the camera and the laser-plane estimation. This step applies a final filtering, where only values between 1 and 10000 mm are assumed to be true.

The laser-finding process is explained in more detail in Algorithm 2.

Algorithm 2 Distance computation for a filtered frame. The full algorithm runs on the CPU. The provided parameters are the filtered frame ($frame$), the minimum luminance for a pixel to be considered ($min_{lum} = 10$), the minimum value a pixel should have to be considered part of the blob ($min_{blob} = 70\%$), the minimum weight for a blob to be considered ($min_{weight} = 25$) and the minimum and maximum distance value for the triangulation to be considered valid ($min_d = c_c/1$ and $max_d = c_c/10000$).

function COMP_DIST($frame, min_{lum}, min_{blob}, min_{weight}, min_d, max_d$)
  for all $i \in COLUMNS$ **do**
    maxidx = 0 ▷ Detect maximum pixel
    for all $j \in ROWS$ **do**
      if $frame(i,j) > frame(i, maxidx)$ **then**
        $maxidx = j$
    if $frame(i, maxidx) < min_{lum}$ **then** ▷ Noise threshold 1
      return $(-1, -1, -1)$
    $blob = min_{blob} * frame(i, maxidx)$ ▷ Determine Blobsize
    $m = n = maxidx$
    while $(frame(i,m) > blob)$ & $(m \in ROWS)$ **do**
      $m = m - 1$
    while $(frame(i,n) > blob)$ & $(n \in ROWS)$ **do**
      $n = n + 1$
    $avg = weight = 0$ ▷ Weighted Average
    for all $j \in [m, n]$ **do**
      $avg = avg + j * frame(i,j)$
      $weight = weight + frame(i,j)$
    $j_{laser} = avg/weight$
    if $avg < min_{weight}$ **then** ▷ Noise threshold 2
      return $(-1, -1, -1)$
    $(x_n, y_n) = CAM\_BACKWARDS(i, j_{laser})$ ▷ Camera projection
    $d = 1 - (a_c * x_n + b_c * y_n)$ ▷ Triangulation
    if $min_d < d < max_d$ **then**
      $z = c_c/d$
      return $z * (x_n, y_n, 1)$ ▷ Distance estimation
    else
      return $(-1, -1, -1)$

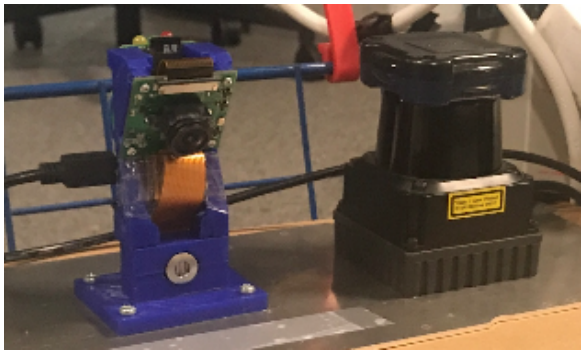


Figure 6.2: Picture of the testsetup. Left the prototyped sensor, right the Hokuyo UTM-30LX , both mounted on a stroller

6.2 Setup

To validate the results of the prototype, its output is compared with the measurements of the Hokuyo UTM-30LX [13]. Both sensors have been mounted on a rigid structure on a stroller, as shown in Figure 6.2. As the measurements of both sensors are in different coordinate frames, transformation matrices have to be estimated.

The reference point for both sensors is defined at the laser source of the prototype, with the z-axis pointing forward and the y-axis upwards. Computing the transformation is done by measuring and estimating the distances and angles manually. For the Hokuyo, the resulting matrix is defined as:

$$\mathcal{P}_{hokuyo} = \begin{bmatrix} 0.9999 & 0.0115 & 109.8701 \\ -0.0115 & 0.9999 & -6.8423 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.3}$$

And for the LaserPi:

$$\mathcal{P}_{camera} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.8520 & 0.5236 & -49.244 \\ 0 & -0.5236 & 0.8520 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

As the prototype computes a 3D point cloud, a 4D projection matrix is used. After projecting the 3D point cloud to the proper reference frame, the y-axis is eliminated. [1]

[1] In an ideal setup, the laser is aligned perfectly with the camera, and hence all y-values should be zero after the projection. Due too the roll of the laser and other small inaccuracies, this may not be the case. But, as the transformation matrix is measured manually, the error in the projection is assumed to be significantly larger than the induced error of disregarding the roll of the laser.

6.3 Post processing

The presented models and computations are able to correct most of the distortions. However, as parts of these models are approximations, there are remaining inaccuracies and residuals. To reduce those inaccuracies, post-calibration is commonly performed [19, 29]. With post-calibration, the output of a prototype and a ground-truth are compared. This difference is then used to make a generalization of the error, which eventually is used to correct the output of the prototype. Both [19] and [29] show that a post-calibration model of $\frac{1}{x}$ is able to correct the distance estimations. After applying the $\frac{1}{x}$ correction, they correct the final residuals with interpolation of lookup tables: these tables state the remaining offset at certain distances.

As the presented system allows the use of a line-laser and a high-distortion lens, a more elaborated (higher dimension), post-calibration model has to be used as opposed to the $\frac{1}{x}$ approximation. Experiments showed that a 2-4 order polynomial produced acceptable results. The result of the post-calibration is shown in Figure 6.3, where different measurements of a straight wall of both the Hokuyo and LaserPi are shown.

Estimation of the polynomial is done by aligning the Hokuyo and LaserPi with a straight wall and recording a set of measurements at an increasing distance. In total 30 sets have been recording at 10cm increases, spanning a distance from 10 to 300cm. Each set holds approximately 5 seconds of data, resulting in 150 measurements of the Hokuyo and 25 of the LaserPi.

To compute the correction, all data points of a single sensor within a set are combined to a superset. Of this superset, the top 5 percent outliers are removed. First, 5 percent of the largest and smallest distance estimations are removed. Of the remaining 90 percent, 5 percent of measurements at each side of the lens boundary is removed as these measurements are most affected by the distortion. In total 81% of a measurement set is used to create a polynomial fit.

Fitting is done with a first-order polynomial for the Hokuyo, whereas the LaserPi is fitted with a second-order polynomial. The difference between these two fits is assumed to be the measurement error. By collecting all differences, a 2-4 order polynomial is estimated, representing the projection error as shown in Figure 6.1.

The computed polynomials are shown in Table 6.1, used to correct the estimated z-value of a measurement using the following equation:

$$z = z - (p00 + p10 * x + p01 * z + p20 * x^2 + p11 * x * z + p02 * z^2 + p21 * x^2 * z + p12 * x * z^2 + p03 * z^3 + p22 * x^2 * z^2 + p13 * x * z^3 + p04 * z^4) \quad (6.5)$$

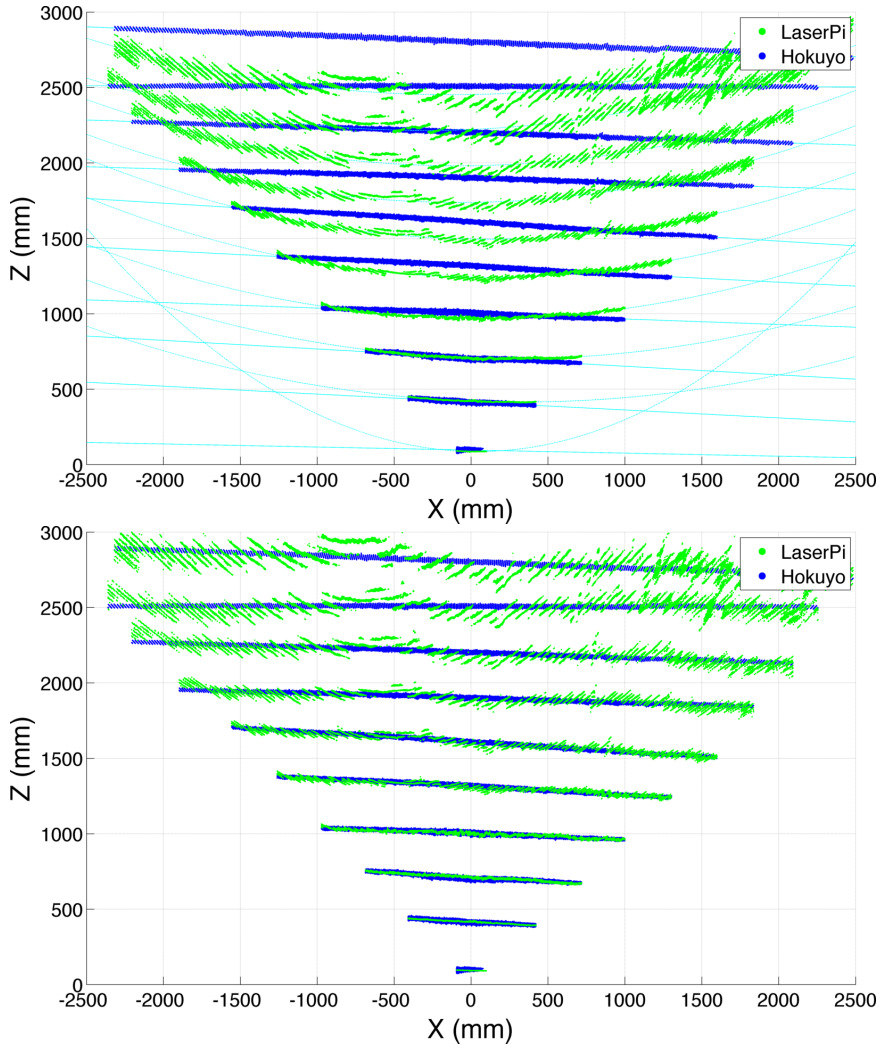


Figure 6.3: Comparison of several sets of measurements between the Hokuyo UTM-30LX and the Prototype (LaserPi), before and after the post calibration step. The cyan coloured lines in the first image indicate the estimated fit per measurement set. The bottom image show the calibrated results, indicating that the performance of the LaserPi is sufficient to use for distances up to 2 meters. Although the post calibration is able to produce proper results, effects of lens distortion in the measurements are still visible in the final projection.

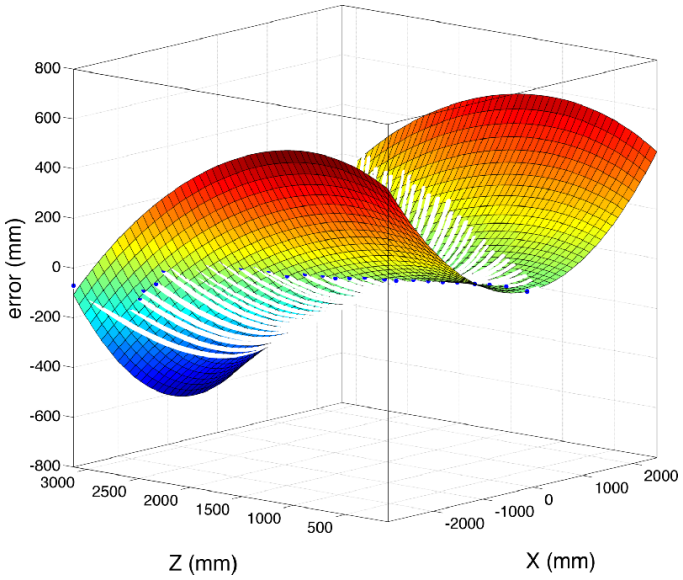


Figure 6.4: Computed error (white) and the 2-4 polynomial post-calibration fit.

p00	-14.2902751197414
p10	0.0050563731787
p01	0.1052658383751
p20	0.0000644780056
p11	0.0000182952860
p02	-0.0001576897638
p21	0.0000000306129
p12	0.0000000017341
p03	0.0000000365886
p22	-0.0000000000076
p13	-0.0000000000001
p04	-0.0000000000052

Table 6.1: Polynomials of the post-calibration fit

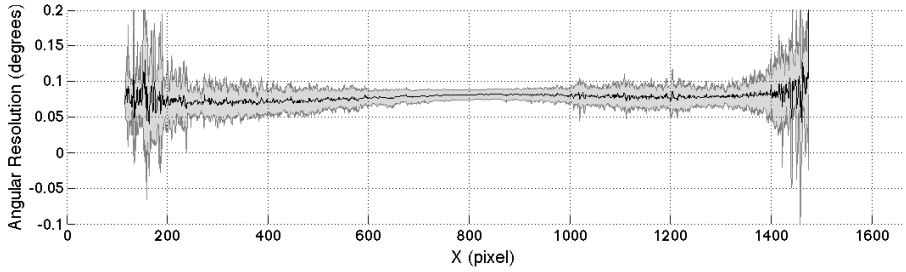


Figure 6.5: Average angular resolution (black) and its standard deviation (grey) per pixel, across the width of the imager. The displayed average contains the error of all measurements. More detailed graphs are shown in Chapter C.

6.4 Results

After applying the post-calibration step, this section shows the final results and analysis of the distance accuracy, (angular) resolution, field-of-view and computational performance. At the end of this section a reflection of the requirements is given.

6.4.1 Distance estimation

Evaluating the distance estimation capabilities of a laser distance sensor is typically done with high- and low- reflectivity objects, in different light conditions. For our setup, we used high and low reflectivity walls indoors and outdoors. The outdoor tests have been performed on a bright and sunny day. Outdoors, the prototype failed to produce any sensible distance estimation, regardless of the tested distance. Therefore, only the test results for the indoor experiments are shown. These are listed in Figure 6.6 and Figure 6.7. More detailed results, including the individual results for each set of measurements, can be found in Chapter C.

The graphs in Figure 6.6 and Figure 6.7 show unfiltered and filtered results. The first shows the error based on all measurements in a set, whereas the latter removes 5 percent of the measurements at the edges. Filtering is done to view the effect of the increasing distortion at lens boundaries.

Results show that both setups have a remaining error: the accuracy is non-zero and varies slightly with the distance, indicating that calibration did not remove all inaccuracies. Additionally, the resolution increases with the distance: at longer distances, it becomes more difficult to estimate the laser reflection, hence higher variance in the measurements will occur. The low-reflectivity setup shows a decrease in resolution at large distances. As the prototype starts failing to detect the low-reflectivity wall, fewer measurements are received, eventually decreasing the field-of-view and therefore also decreasing the variance.

In Figure 6.5 the average angular resolution per pixel position is shown. Due too the distortion, projection, and correction in our setup, a slight variance can be observed.

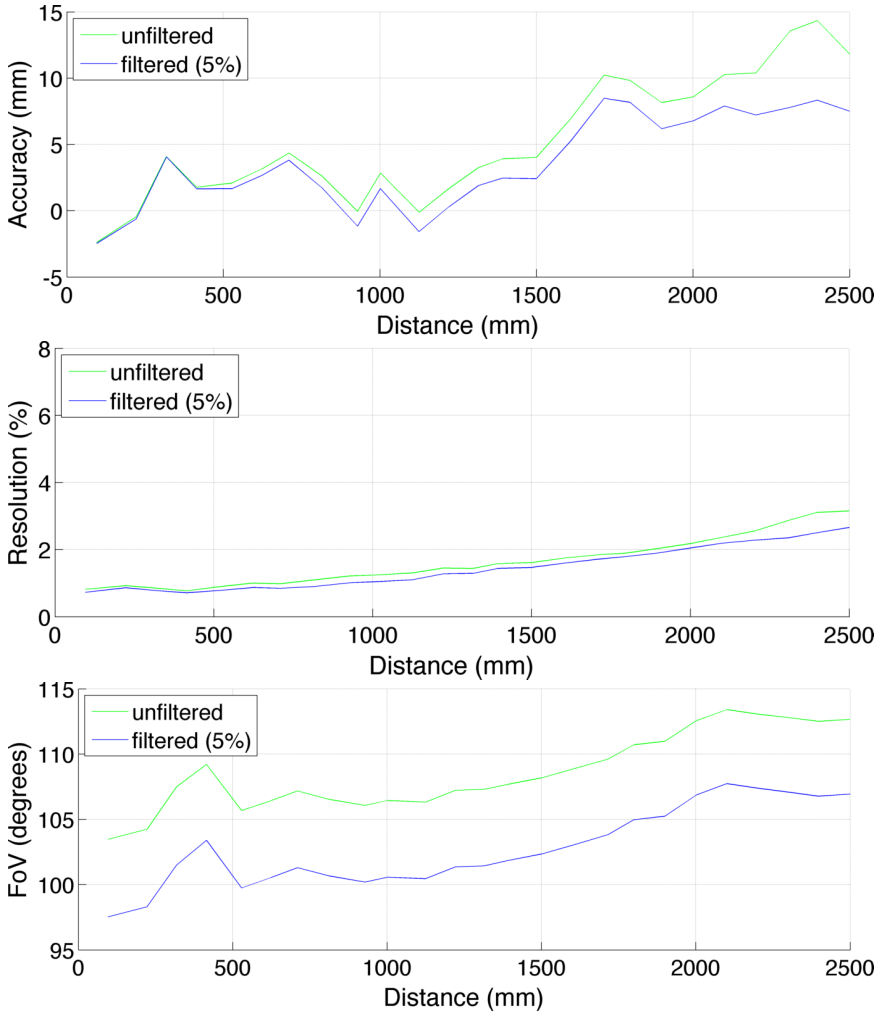


Figure 6.6: Summarised indoor testresults of a high reflective wall. The top image shows the (average) remaining error with the computed ground truth. In the middle image the error (standard deviation) of the measurements with respect to the measured distance are displayed. The bottom image shows the observed field-of-view.

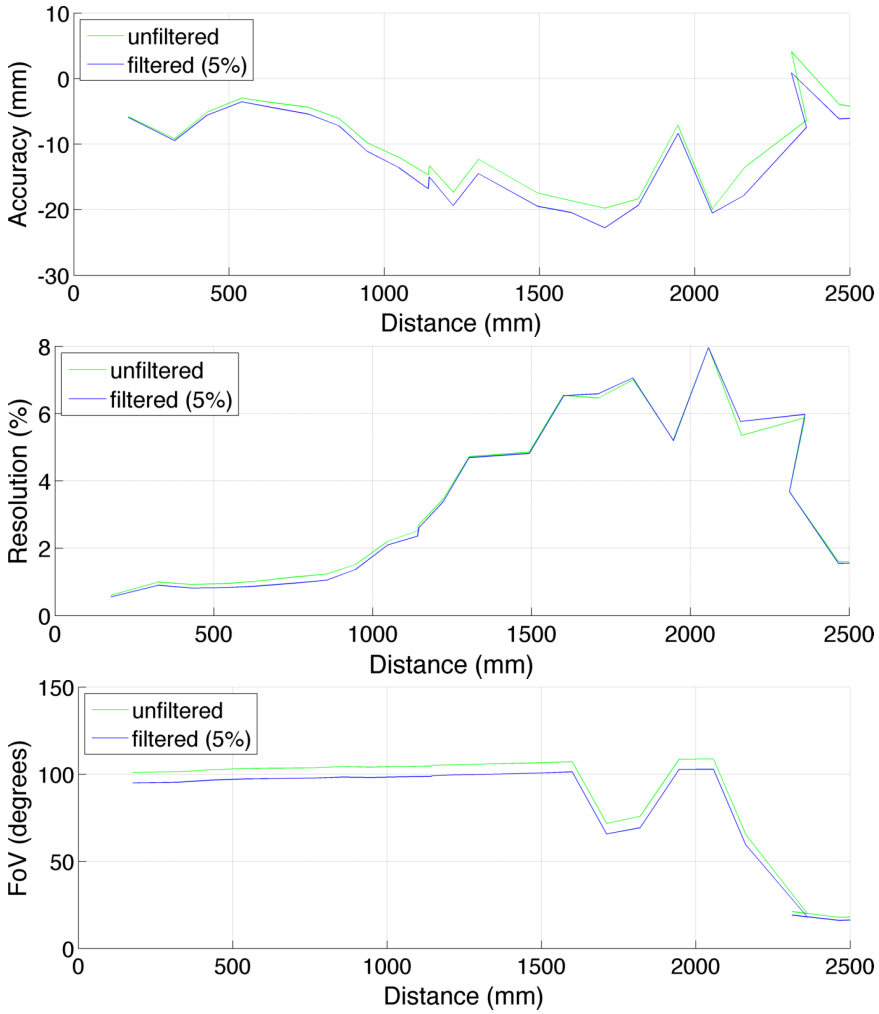


Figure 6.7: Summarised indoor test results of a low reflective wall. The top image shows the (average) remaining error with the computed ground truth. In the middle image the error (standard deviation) of the measurements with respect to the measured distance are displayed. The bottom image shows the observed field-of-view.

	Average
CPU	50Mb
GPU	21Mb
Total	71Mb

(a) Measured memory usage (RAM) of the LaserPi. The Raspberry Pi has a maximum of 512Mb available. In our prototype this memory is split 50/50 between the CPU and GPU.

	Average	Min	Max
CPU	22.57 ± 5.45	15.81	103.58
GPU	66.87 ± 7.72	52.96	111.22
Total	89.42 ± 10.84	70.15	199.16

(b) Average processing time (ms) of a single frame, based on a period of 20 minutes. Estimates are captured by retrieving the GPU time upon start and finish of the CPU image processing part. The start time is determined by using the presentation timestamp of a frame, provided by the GPU as described in Chapter 5.

Table 6.2: Performance measurements of the LaserPi

This variance is largest at the edges of the lens. The average angular resolution is estimated to be around 0.8 degrees.

6.4.2 Performance

To provide some insight into the processing capabilities and limitations of the LaserPi, several performance measurements have been performed. For all measurements, the prototype was producing results at 10Hz. An overview of the memory usage and the processing time is given in Table 6.2.

Viewing the Linux-provided running CPU statistics showed that the main script had a utilization of around 35 percent, with the five- and fifteen- minute load showing an average of 0.50 and 0.35 respectively[2]. It should be noted that the load-estimates include the processing time to maintain a connection, display information, write statistics to file and maintain the overall operating system.

Regarding the GPU, only an indication of the load of the dual-core Vector Processing Unit (VPU) could be estimated, being idle >80 percent of the time. The 3D processing pipeline used by OpenGL (using the 24GFLOPs quad-core QPU), does not allow benchmarking. However, as the VPU controls the QPU, it is assumed that the QPU isn't running at full load either. Additionally, with the GPU spending on average only 67ms on a frame, and the imager producing frames at 10Hz, the estimated GPU load is 0.67. The discrepancy between the measured idle-time and the estimated load might be caused by synchronization and waiting of threads and hardware-blocks.

[2]The final statistics have been captured after a 20 minute run of the prototype

Description	Required	Measured	RPLiDAR [5]
Mechanics	Solid-State	Solid-State	Rotating
Cost @ 1pcs	-	$125.6	$450
Cost @ 1000pcs	<$30	$35 + ($25)	$\sim$ 200
Dimension (max)	10x4x3 cm	8.5x4.5x3.5 cm	7.6x7.6x4.1 cm
Resolution	$<$ 1% @ 2m	$\sim$ 1.85% @ 2m	$\sim$ 1% @ 2m
Field-of-view	120 deg	105 deg	360 deg
Angular Resolution	<0.9 deg	$\sim$ 0.8 deg	$\sim$ 0.9 deg
Maximum distance	$>$ 2m	$\sim$2.5m	<6m
Update Rate	>10 Hz	10 Hz	<10 Hz
Samples	$>$ 1333/s	16640/s	4000/s
Eye Safe	Class 1	Class 1 (>0.8cm)	Class 1
Processing	Fully Embedded	Fully Embedded	Fully Embedded

Table 6.3: Overview of the requirements and how the prototype matches. Details on the cost-estimates are given in Chapter B. Comparing the results of the first prototype and the RPLiDAR shows competitive specification at a lower cost, while allowing improvement of several properties.

6.4.3 Requirement evaluation

Table 6.3 shows how the prototype matches the requirements. While dimension, cost, field-of-view, and resolution do not meet the minimum, performance analysis has shown that there is still enough capability in the BCM2835 left to improve. Furthermore, all of these (measured) values are an initial estimate. Cost can be reduced by more investigation and negotiation with suppliers. The dimensions can be reduced by using a different construction, with an estimate for a second prototype having a dimension of around to 8x3.5x3 cm. Field-of-view can be increased by using a different laser, as the used module turned out to have a physical limit of 115 degrees.

Reviewing the accuracy and the resolution estimates of Table 4.10, it is suggested that the current image processing pipeline has a sub-pixel accuracy of just below 0.4 pixels: at 2 meters, both experiments and theorised model show a resolution of $\sim$ 1.8 percent and at 2.5 meters, both estimate a resolution of $\sim$ 2.4 percent.

Chapter 7

Conclusion and Future Research

7.1 Conclusion

This thesis questioned the possibility of improvement of laser distance sensors for consumer robotics. It was shown that existing models and literature are not able the satisfy requirements such as a solid-state design, low-cost and large field-of-view. To meet these constraints, a generic triangulation model, independent of camera properties, has been introduced.

The presented model works by assuming the normalized position of a laser reflection, acquired from the backward projection of a camera model. With the normalized position and a plane representation of the (line) laser projection, the triangulation model is able to estimate a 3D position. The addition of a background irradiance model and laser safety standards eventually allows prediction of the distance estimation limitations of a prototyped sensor.

After theorized optimisation of the model parameters, a prototype has been proposed, build and calibrated. Camera calibration was done using existing tools, with the addition of capturing the laser reflection in the calibration images. This reflection has been used to correct for misalignment errors between the camera and laser, resulting in an integrated calibration process. That is, parameters and errors of both the camera and triangulation models where estimated using the same data set.

To ensure eye-safety and maximum detection distance, a phase lock loop mechanism has been designed for synchronizing the laser activation and the imager exposure. Using the GPU clock, the phase lock loop was shown to be able to synchronize the periods of the imager and laser with an accuracy of several tens of microseconds.

Benchmarking the sensor showed that accuracies similar to existing low-end systems can be reached while having a fraction of the cost. Additional analysis showed that the current implementation has a low processor load for both the CPU and GPU, allowing improvements and more advanced processing steps in future setups.

In conclusion, this thesis showed that it is possible to reduce the cost of current laser distance sensors, while maintaining or improving the performance, by developing a prototype in an integrated approach.

7.2 Future research

This section shows several shortcomings and ideas for improvement of the performance of the prototyped sensor and the derived models.

7.2.1 Sensor model

Regardless of all theorized corrections, the prototype (and hence the proposed sensor model) still produce slightly erroneous distances estimations. Even after post-calibration, the results are not error-free. Part of this could be devoted to improper camera calibration or selecting an improper camera model: the results show a large amount of uncorrected-distortion at the edges. Additionally, the post-calibration fit uses a 2-4 polynomial, whereas existing literature applies a $\frac{1}{x}$ correction. By collecting more data, improved analysis and fitting of the post-calibration error could be done.

The selected plane estimation has some limitations too. It only works when the optical axis of the camera and the laser are not are aligned parallel or perpendicular to each other. The former would result in $b_c = 0$, and hence the depth information from an image (y-position of a pixel) is lost (see (3.17)). In the latter case, b_c will be estimated as ∞, which would also remove the depth information of an image.

The current model also does not include the scheimpflug condition [31, 30], commonly used in triangulation setups to increase the depth-of-field. While the utilized camera model can correct for tangential errors, it is only able to correct for small deviations.

It might also be noted that the Spectra-estimates are not correlated with the imager, while definitions of for example the quantum efficiency of a pixel are explained. Although it should be possible to correlate the background irradiance with the effect it has on the digitized signal of a pixel, many conversions, specifications and deep knowledge of the imager is needed. As such, this correlation is not included in the model, but it could help improve the limit estimates of a hypothesized sensor.

Additionally, the Spectra-estimates have not been verified with measurements during testing of the prototype. While the estimation results indicate proper limit estimations, verifying the Spectra tables and the output of the sensor might improve the prediction.

In this thesis, laser specification and limitations where presented, yet, actual intensity-measurements have not been performed. To guarantee eye-safety of the final prototype and proper deduction of all safety regulations, measurements need to be taken

7.2.2 Calibration

With respect to calibration, there are several improvements possible. Foremost, the used toolbox for camera calibration uses an optimization step to find the exact corner of the squares in the checkerboard. With the laser pointing at the checkerboard, the optimization sometimes optimises towards the laser reflection, as those pixels are much brighter and hence assumed to be white pixels. As such, small errors in the camera intrinsic estimates are introduced. Using larger checkerboard patterns, alternating the laser projection or updating the optimization step might solve this challenge.

Additionally, the laser-plane estimation requires a user to select the laser reflection manually, which consumes time and is error-prone. Just as the camera calibration, this step could also be automated. When automated, the linear optimization step, which was proposed in this thesis, would not be needed anymore.

The linear optimization step in itself also introduces errors in the model: while optimising the global plane, it does so by updating the selected pixel coordinates of all images, regardless of how well they represent the center of the laser reflection.

The utilized post-calibration corrects for errors in depth-estimation. Stated differently, it only corrects for distortion in the vertical dimension of the imager. As the lens distortion affects both the horizontal and vertical dimension, residual errors might exist across the field-of-view. Additional research needs to be applied to determine the severity of this limitation.

When the post-calibration step is updated to include the horizontal distortion of the lens, the triangulation-estimation step could possibly be discarded. As the post-calibration is used to correct both depth and field-of-view in a non-linear fashion, this correction might be able to correct for misaligned triangulation parameters. Using the theorized triangulation parameters and post-calibration fit, the full calibration process could potentially be simplified.

7.2.3 Synchronisation

Currently, the phase lock loop does not correct for any timing between the exposure of the imager and receiving the first pixels at the GPU. The presented mechanism just assumes that this happens instantly. Measuring this offset might increase the accuracy and hence increase the captured laser reflection intensity.

Another shortcoming of the taken approach is that the phase lock loop does not account for any clock drift between the hardware PWM and the GPU clock. As such, the provided solution is essentially an open loop system. Eventually, the laser and imager might go out of sync. Accounting for clock drift could be done by allowing the hardware PWM to produce interrupts at the GPU. These interrupts could also eliminate the restart-sequence to estimate the start time of the PWM signal, possibly decreasing the system error even more.

In the current implementation, the CPU controls the framerate at every frame update. Ideally, after settling, the controller should be able to relax the number of corrections, reducing any overhead on the GPU. To do such a thing, more appropriate controllers might be available, as the presented regulator only consists of a P-gain.

7.2.4 Image processing

Our prototype uses a straightforward image filtering for which no thorough testing or optimisations have been performed. As a result, the filters have difficulty in finding the laser reflection at black objects or in bright environments. Additionally, the corrections of the wall-measurements show curved projections, while these distance estimates ideally should produce a fluent straight line.

By testing different filter parameters, the sensitivity of the laser-dot localization can be enhanced. Improved detection in bright environments could also be established by using data produced by the ISP or imager such as the white balance of the taken frame: bright environments typically have higher white-counts.

Tests also showed that estimates at small distances are affected by ghosting: that is, at short distances the estimation is affected by the indirect reflection of the laser light. More advanced filters could take this into account. For example, different filters could be assigned to different pixel location: correcting for ghosting at short distance and filters with enhanced sensitivity for the laser reflection at larger distances.

Besides filtering, the laser-pixel position estimation could be improved too. Currently, a blob detection algorithm is used which assumes a uniform reflection across an object. Since the blob boundaries are only determined by a threshold, patterns or nonuniform reflections might result in improper boundary estimation. Existing literature shows that precise laser-dot localization can be achieved by modeling the Gaussian shape of the laser beam. Applying such a technique might improve the sub-pixel accuracy and hence the distance estimation.

In our prototype, the laser detection is run at the CPU, whereas it was shown that the GPU is not under full load yet. Pushing the dot detection to the GPU might decrease the total processing time of a single frame, allowing a higher update rate.

7.2.5 General

The used imager (and the ISP) have the highest cost of the prototype ($40 at $64, including licensing). Recently, direct communication with the CSI-2 peripheral became available on the Raspberry Pi platform. As such, different imagers can now be used from which the raw (Bayer) data can be retrieved, allowing the usage of monochrome or global shutter imagers. Using a monochrome imager would increase sensitivity, as pixels are not extended with color filters. A global shutter imager would result in a reduction of the pulse width of the laser. This, in turn, would allow a more powerful laser and hence result in an increase of the maximum detection distance. Selecting a different imager would also allow a significant reduction in the cost of the setup. It should, however, be noted that reading the raw data from the CSI-2 peripheral bypasses the ISP, so the pixel correction and balancing operations of the ISP are not applied.

Throughout all experiments, the current mechanical setup showed to be prone to movement. Further research could be dedicated to developing a more rigid construction, resulting in more stable measurements.

To make the sensor suitable for, for example, the drone industry, tests, and estimates of power consumption need to be made. With the current performance analysis showing that the SoC only has a load of 0.35, frequency reduction or other power controlling actions could be taken. This however also requires a mapping of the power consumption under the current load.

Bibliography

[1] BCM2835 ARM Peripherals. Datasheet Version CB4 0WW, Broadcom Europe Ltd., 2012.

[2] Multi-media abstraction layer (mmal). Reference Manual Version 0.1, Broadcom Europe Ltd, 2012.

[3] VideoCore IV 3D Architecture Reference Guide. Datasheet Version VideoCoreIV-AG100-R, Broadcom Corporation, September 2013.

[4] IMX219PQH5-C. Datasheet Version E13Y12G47, Sony Corporation, November 2016.

[5] RPLIDAR A2, Low Cost 360 Degree Laser Range Scanner, Introduction and Datasheet. Datasheet, Shanghai Slamtec Co., Ltd, April 2016.

[6] User's Manual and Technical Specifications. Sweep v1.0. Scanning Laser Range Finder. Datasheet, Scanse LLC, July 2017.

[7] Velodyne LiDAR, HDL-64E, high definition real-time 3D LiDAR. Datasheet, Velodyne LiDAR, Inc., 2017.

[8] LMS1xx, Compact and Economical, Even in Harsh Environments. Datasheet, SICK AG, 2016.

[9] Anas Alhashimi. *Statistical Calibration Algorithms for Lidars*. PhD thesis, Lulea University of Technology, August 2016.

[10] Standard Tables for Reference Solar Spectral Irradiances: Direct Normal and Hemispherical on 37° Tilted Surface. Standard, ASTM International, 2012.

[11] Bryce E. Bayer. Color imaging array. Patent US3971065 A, July 1995.

[12] Jean-Yves Bouguet. Camera calibration toolbox for matlab. Online, June 2017. url:http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

[13] Scanning Laser Range Finder, UTM-30LX/LN, Specification. Datasheet, Hokuyo Automatic Co.,Ltd, November 2012.

[14] How to Evaluate Camera Sensitivity, Comparing camera performance using the EMVA1288 imaging performance standard. White paper, Shanghai Slamtec Co., Ltd, August 2016.

[15] L. M. Eriksson and M. Johansson. Simple PID tuning rules for varying time-delay systems. In *2007 46th IEEE Conference on Decision and Control*, pages 1801–1807, December 2007.

[16] Margaret Fleck. Perspective projection: The wrong imaging mode. Technical Report 95-01, Department of Computer Science, University of Iowa, 1995.

[17] G. Fu, P. Corradi, A. Menciassi, and P. Dario. An integrated triangulation laser scanner for obstacle detection of miniature mobile robots in indoor environment. *IEEE/ASME Transactions on Mechatronics*, 16(4):778–783, August 2011.

[18] Joseph D. Gaeddert. Writing a phase-locked loop in straight c. Online, November 2013. url:http://liquidsdr.org/blog/pll-simple-howto/.

[19] Jason Gao and Li Shiuan Peh. A smartphone-based laser distance sensor for outdoor environments. In *International Conference on Robotics and Automation (ICRA)*, pages 2922–2929, 2016.

[20] Abbas Emami-Naeini Gene F. Franklin, J. David Powell. *Feedback Control of Dynamic Systems*. Pearson, 2010. ISBN:9780135001509.

[21] J. Heikkila. Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1066–1077, October 2000.

[22] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112. IEEE, 1997.

[23] Guan-Chyun Hsieh and J. C. Hung. Phase-locked loop techniques. A survey. *IEEE Transactions on Industrial Electronics*, 43(6):609–615, December 1996.

[24] IEC 60825-1: Safety of laser products - Part 1: Equipment classification and requirements. Standard, International Organization for Standardization, May 2014.

[25] Jan Stroeken Jaap Schrage, Hans van Daal. *Regeltechniek voor HTO*. HBuitgevers, 2005. ISBN:9789055746248.

[26] Zhao Jianlin, Hao Jianhua, Li Enpu, Guo Wenge, and Yang Dexing. A method of improving ccd's resolution in laser triangulation measurement. *Acta Photonica Sinica*, 26(11):998–1002, January 1997.

[27] Dave Jones. Picamera. Reference Manual Version 974540f3, Waveform, May 2017.

[28] J. Kannala and S. S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1335–1340, August 2006.

[29] K. Konolige, J. Augenbraun, N. Donaldson, C. Fiebig, and P. Shah. A low-cost laser distance sensor. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3002–3008, May 2008.

[30] A. Legarda, A. Izaguirre, N. Arana, and A. Iturrospe. A new method for scheimpflug camera calibration. In *2011 10th International Workshop on Electronics, Control, Measurement and Signals*, pages 1–5. IEEE, June 2011.

[31] H Louhichi, T Fournel, JM Lavest, and H Ben Aissia. Self-calibration of scheimpflug cameras: an easy protocol. *Measurement Science and Technology*, 18(8):2616–2622, April 2007.

[32] Inc. Luminar Technologies. Luminar. Online, June 2017. url:https://www.luminartech.com/.

[33] C. Mertz, J. Kozar, J. R. Miller, and C. Thorpe. Eye-safe laser line striper for outside use. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 507–512 vol.2, June 2002.

[34] Kenro Miyamoto. Fish eye lens. *Journal of the Optical Society of America*, 54(8):1060–1061, August 1964.

[35] Manesh V. Mohan, S. Anjana Devi, C. H. Teena, and Anu Abraham. A method for minimum range extension with improved accuracy in triangulation laser range finder. In *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*, pages 115–121, New York, NY, USA, 2011. ACM.

[36] JF Randall and J Jacot. Is am1. 5 applicable in practice? modelling eight photovoltaic materials with respect to light intensity and two spectra. *Renewable Energy*, 28(12):1851–1864, 2003.

[37] Manan Raval, Christopher V. Poulton, and Michael R. Watts. Unidirectional waveguide grating antennas with uniform emission for optical phased arrays. *Optics Letters*, 42(13):2563–2566, July 2017.

[38] S. Russo, K. Harada, T. Ranzani, L. Manfredi, C. Stefanini, A. Menciassi, and P. Dario. Design of a robotic module for autonomous exploration and multimode locomotion. *IEEE/ASME Transactions on Mechatronics*, 18(6):1757–1766, December 2013.

[39] C. Schlegel, T. Hassler, A. Lotz, and A. Steck. Robotic software systems: From code-driven to model-driven designs. In *International Conference on Advanced Robotics*, pages 1–8, June 2009.

[40] W. Schottky. Über spontane stromschwankungen in verschiedenen elektrizitätsleitern. *Annalen der Physik*, 362(23):541–567, 1918.
[41] Richard Szeliski. *Computer Vision: Algortihms and Applications*. Springer, 2011.
[42] OpenCV team. Open source computer vision library. Online, June 2017. url:https://opencv.org/.
[43] Michel Thoby. About the various projections of the photographic objective lenses. Online, 2012. url:http://michel.thoby.free.fr/Fisheye_history_short/Projections/Various_lens_projection.html.
[44] S3 Solid State Lidar Sensor. Datasheet, Quanergy Systems, Inc., 2016.
[45] Yuan-Jay Wang Ying J. Huang. Robust pid controller design for non-minimum phase time delays systems. *ISA transactions*, 40(1):31–39, 2001.

Appendix A

Camera: Projection, Distortion and other Properties

The camera is a vital part of the triangulation setup. This chapter introduces the basic principles of image projection and frame capturing of a camera system and yields as background information and reference material for this thesis. The principles explained here are assumed to be known and will be directly or indirectly used throughout this thesis.

Before the inner details of a camera can be explained, we first need to provide some used definitions. In this thesis, a camera is defined to be a mechanical device consisting of a lens and an imager. The former bends and shapes passing light rays, such that a projection with specific properties is created. The latter is an array of light-sensitive elements, known as pixels, able to electronically encode the amount of light hitting a discrete surface.

Using these definitions, we explain in the next sections how the properties of pixels, imagers, and lenses affect a captured image.

A.1 Pixels

A pixel is a small light-sensitive element, which is able to transform an incoming light ray (or photon) into an electric signal. A schematic overview of its components is shown in Figure A.1.

At the top of the image, several photons hit the light-sensitive part (sensor) of the pixel. The sensor converts the incoming photons into electrons which in turn are stored in the pixel well. As the well gradually captures more electrons over time, its electrical potential increases. This potential is eventually measured and represents the pixel signal. The amount of time that a well collects electrons of which the pixel signal is generated is called the exposure time. When the exposure time is such that the maximum capacity of the well is reached, the pixel is said to be saturated and no more electrons can be captured.

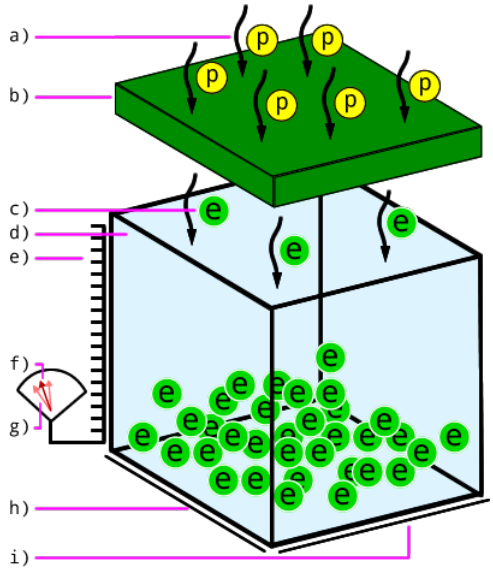


Figure A.1: Schematic overview of the internals of a CMOS pixel. a) Incoming photons. b) Sensor. c) Excited electrons. d) Pixel well. e) Saturation capacity (pixel height). f) Pixel signal. g) Temporal dark noise. h) Length of pixel. i) Width of pixel.

Using the EMVA1288 standard [14], a more concise and elaborate description can be given, providing insight into how all these components work together.

Starting at the top of Figure A.1, six photons seem to reach to the sensor at the same time. In reality, each photon might have a different time of arrival. If a pixel needs to operate in a dark environment, has a small size or has to work with short exposure times, these differences in arrival times introduce noise: some pixels might receive photons, while others don't. When analyzing the arrival time of photons in 1918 [40], it became apparent that the arrival time of photons could be described by a Poisson process. The introduced noise was called Shot Noise, ϵ_N, and can be computed by the square-root of the number of photons, N:

$$\epsilon_N = \sqrt{N} \tag{A.1}$$

It is however not very common to describe the shot noise of a pixel per photon. Instead, the light intensity (number of photons per surface) is used. If the surface area of a pixel is described with P and the light intensity as L, the shot noise can be expressed as:

$$\epsilon_N = \sqrt{P * L} \tag{A.2}$$

From Figure A.1 it can also be deduced that not all received photons result in the release of an electron, e^-. The efficiency of conversion $\frac{N}{e^-}$ is called the quantum efficiency (QE) and depends on manufacture properties as well as the color (wavelength) of the

photon. For simplicity, we assume in this introduction that the QE for all wavelengths is the same. Under that assumption, the efficiency of a pixel can be defined as:

$$\eta = \frac{N}{e^-} \tag{A.3}$$

Hence, the number of electrons, N_e, and the shot noise, ϵ_e, can be computed as:

$$\begin{aligned} N_e &= N * \eta = P * L * \eta \\ \epsilon_e &= \sqrt{N * \eta} = \sqrt{P * L * \eta} \end{aligned} \tag{A.4}$$

Since the maximum number of electrons in a well can be determined, the pixel signal, s, becomes:

$$s = min(N_e, sc) = min(L * P * \eta, sc) \tag{A.5}$$

where sc is the saturation capacity of the pixel.

Due too limitations in the process of manufacturing pixels, imperfections in the readout mechanism are introduced. This reading error is called temporal dark noise (TDN). Combining it with the shot noise, the total noise in a pixel signal can be described with:

$$n = \sqrt{(TDN)^2 + (N_e)^2} = \sqrt{(TDN)^2 + (L * P * \eta)} \tag{A.6}$$

From which we can determine the (linear) signal-to-noise ratio:

$$SNR_L = \frac{s}{n} \tag{A.7}$$

or in decibels:

$$SNR = 20log_{10}\frac{s}{n} \tag{A.8}$$

Using these equations, the effect of pixel size and dark environments on a pixel signal and its noise can be analyzed. In Table A.1 the parameters of three imagers are shown. With the previously introduced equations, Figure A.2 can be constructed, which shows the signal and noise values of these imagers versus the light density.

Looking at Figure A.2 it can be observed that each sensor produces a line with a different slope. This slope indicates the rate at which the number of electrons in a well increases with respect to an increase in photons: the steeper the line, the faster the pixel reaches its saturation point. In other words, the slope represents the pixel sensitivity. The sensitivity of a pixel is commonly expressed as luminous exposure, or $mV/Lux * second$, which includes the QE of all receivable light colors (wavelengths).

The inset of Figure A.2 shows also the point at which a pixel produces a signal which is higher as its noise ($SNR \geq 1$) and hence can be differentiated. As can be observed, there is no direct correlation with the sensitivity: while the ICX414 (green) is more

Sensor	Pixel Size (μm)	η (%)	TDN ($e-$)	sc ($e-$)
ICX618 (CCD)	5.6x5.6	70	11.73	14508
ICX414 (CCD)	9.9 x9.9	39	19.43	25949
IMX249 (CMOS)	5.86x.586	80	7.11	33105

Table A.1: Overview of specifications of three different image sensors. This information is copied from [14].

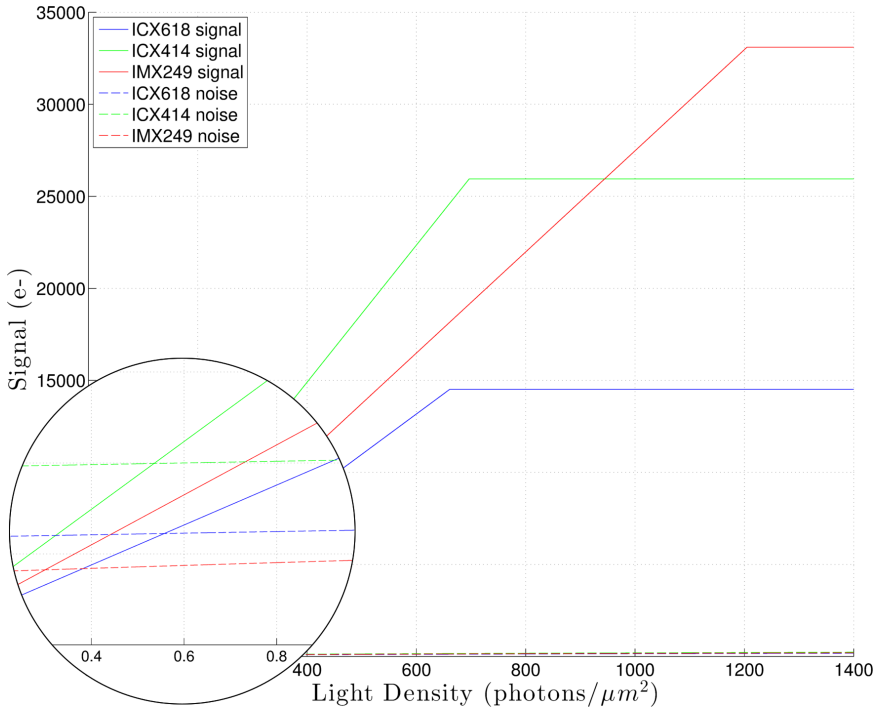


Figure A.2: Signal and Noise versus light density. The inset shows the point at which the imager produces a signal which can be differentiated from the noise.

sensitive than IMX249 (red), the ICX414 requires a higher pixel count, and therefore more light, before a detectable signal is measured.

Comparing the point of signal detection with the moment at which the pixel saturates provides the working area of the imager, or its dynamic range (DR). It provides information when a pixel is said to be black (when $SNR = 1$) or white (fully saturated, when the signal is at its maximum). The maximum SNR is determined by computing the difference between the signal and noise value at the moment of saturation.

While the signal seems to be only dependent on the light density, it should be noted that the exposure time has an effect too. In the Figure A.2 the number of photons reaching the pixel is normalized over the pixel size (hence light density), independent of the time at that a pixel it is exposed. By using longer exposure times, more photons reach the pixel and hence, producing a 'higher density', resulting in an increased signal as opposed to short exposure times. When the exposure time is really short additional photons should be brought into the environment, commonly done by utilizing a flash.

A.2 Imager properties

Arranging pixels in a 2D grid produces an imager. As different applications might require different functionalities of an imager, they commonly have a variety of settings. This section shows the effects of relevant properties.

A.2.1 Monochrome and color

When identical pixels are arranged in a grid, a black (monochrome) image is produced. In order to capture color images, Bayer [11] developed in 1975 the Bayer pattern. By placing red, green and blue filters over the sensitive layer in a pixel, Bayer was able to reconstruct color images. The used pattern is shown in Figure A.4. Utilising this pattern, a (raw) color image from an imager might look like Figure A.4a:left, after which Figure A.4a:right is produced by demosaicing. Demosaicing is the process of interpolating (or averaging) the individual pixels in such a way that each pixel can be described with an RGB value. Modern systems contain typically a specialized image signal processing (ISP) to perform such a task. The ISP commonly also applies other image correcting techniques such as color correction and noise reduction.

A.2.2 Cropping and binning

When an imager is built it has a fixed resolution: the physical amount of pixels in either dimension cannot be changed. In some applications, the maximum resolution of an image is not needed. To reduce the amount of processing (or load of the ISP) functions such as *cropping* and *binning* have been developed. Figure A.4b and Figure A.4c show the effect of these functions respectively.

Cropping allows selection of a subset of pixels from the imager. In Figure A.4b, for example, the cropped image is half the size of the full image, while providing equivalent details. Binning (Figure A.4c) works by averaging pixels. With a 2-binning process, the

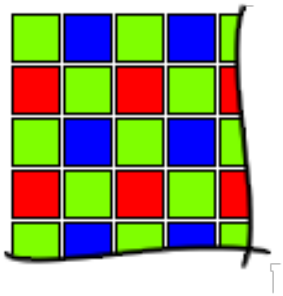
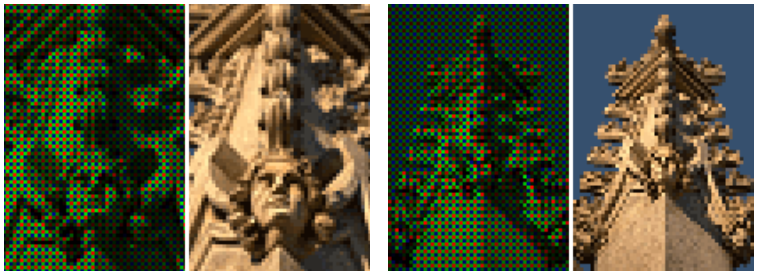


Figure A.3: The Bayer Pattern. Each pixel has a dedicated color filter. As the human eye is more sensitive to the color green, twice as many pixels have a green filter. [11]

(A.4a) Captured image, raw and after processing.



(A.4b) Cropping from the center (A.4c) 2-Binning

Figure A.4: Raw and processed images. The bottom images show the effect of cropping (left) and binning (right), which are both half of the original resolution.
Top images are courtesy of `http://www.cambridgeincolour.com/tutorials/camera-sensors.htm`

value of a pixel in the final (raw) frame is based on the average of 2 pixels from the imager. It allows utilization of a large area of the imager, decreases noise effect due too the averaging and improving bandwidth usage at the cost of lesser details.

A.2.3 Exposure and shutters

Having specified the internals of a pixel and how pixels together form an image, the mechanisms of reading all these pixels values can be discussed. More specific, this section explains the effect of exposure time, framerate, shutter time and how all these cooperate.

To start, all (CMOS)-imagers generally can be classified into two types of image retrieval: they are either rolling-shutter or global-shutter. The first works by capturing (or storing/buffering) individual rows from of the imager sequentially, whereas the latter buffers all rows at the same time. While rolling shutter imagers tend to be cheaper, their approach to image retrieval has several effects on timing of the exposure as well as distortion. To give a better insight, Figure A.5 and A.6 show a schematic overview of the image capturing process of both shutter types.

Each row of pixels contains a reset and store trigger. If the reset is active, the charge (well) of each pixel in the row is cleared. When the store trigger is activated, the pixel signal is read and stored in the buffer. At the top of both Figure A.5 and A.6 the time schedule of the reset and store signal are shown for a duration of 35 ticks. In this period two frames of 3x3 pixels are captured. Both imagers have a *framerate* (or in this case: period) of 18 ticks: that is, every 18 ticks the imager is triggered to capture a new frame. The shutter time (the time between reset and store) is 9 ticks and the time to read and store a row of pixels takes 6 ticks. Below the timeline, at annotated ticks, the state of the imager and its buffer (also 3x3 pixels) are shown. The number within a pixel represents the photon/electron count (QE = 1). Grey pixels contain irrelevant data, whereas white pixels are used in a final frame. The colored arrows indicate activity: a line is receiving a reset or store signal, or a line is being read/transferred to the buffer. The small yellow circle at the corner of the imager indicates the position of a light source. As can be observed at tick 19, the light source changes position from the top left corner to the bottom right. At the bottom of the image, the final captured frames are shown.

The examples in Figure A.5 and A.6 only mention the term shutter time and not the commonly used term exposure time. This is because there is an important difference between both:

- The *shuttertime* of an imager is the time between the *reset* and *store* triggers of a pixel. In both figures, the shutter time equals 9 ticks.
- The *exposuretime* of an imager is the time between the start of the reset of the first pixel and the time that the last pixel of the full imager is read. For Figure A.5 this equals 27 ticks (3x6 ticks to read a line and 1x9 for the initial exposure, whereas the exposure time for the global shutter equals 18 ticks (1x6 ticks to read, as all lines are read at the same time, and 1x9 for the initial exposure).

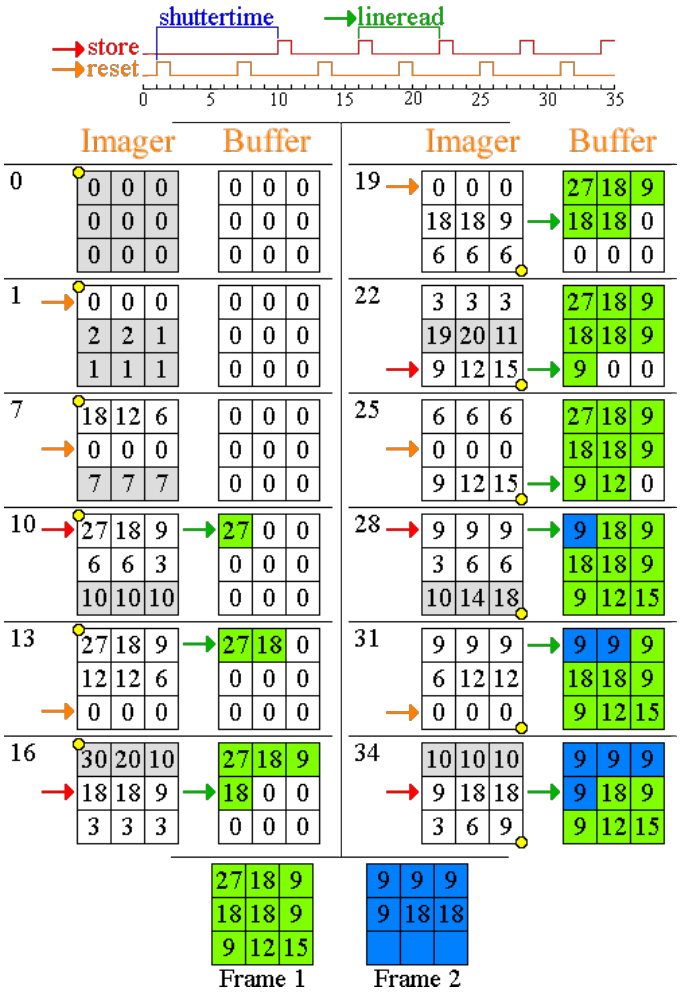


Figure A.5: Simplified schematic functioning of a rolling shutter imager. Notice that as the light source changes position at tick 19, the bottom row of the imager is still being exposed. As a result, the final green frame contains a 'blurred' last row due to the effect of partial exposure. Also, note that the 'final' blue frame is still missing a line after 35 ticks because at that moment the missing row is not yet pushed to the buffer.

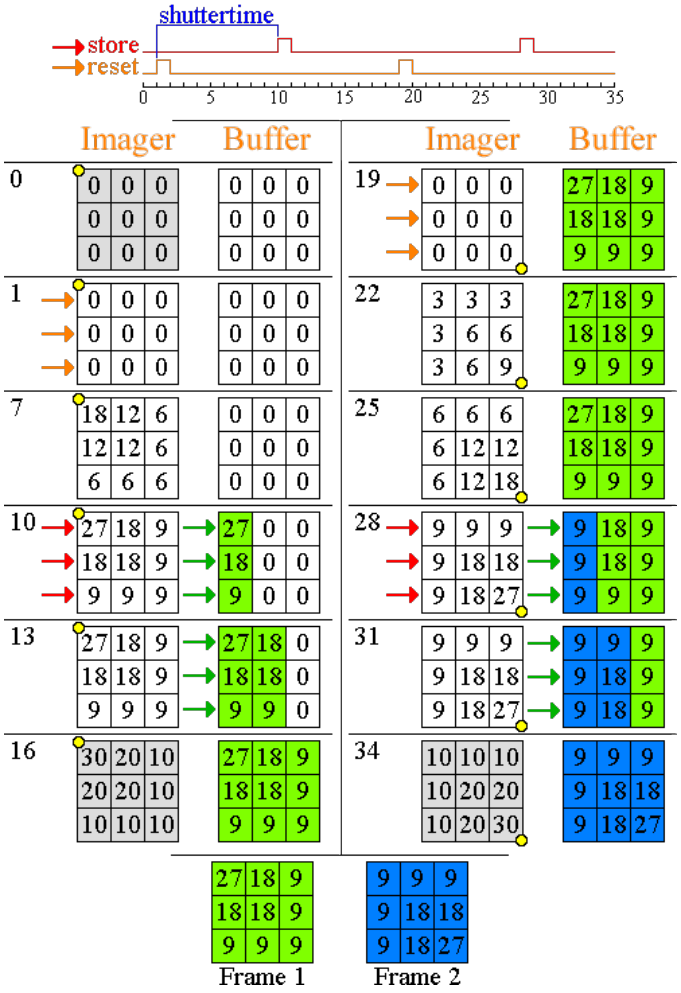


Figure A.6: Simplified schematic functioning of a global shutter imager. Notice that while the light source change position at tick 19, no disturbances are captured in the final frame.

	Rolling shutter	Global shutter
exposure	line by line	full imager at once
exposuretime	(#rows * readout) + shuttertime	readout + shuttertime
distortion	severe by fast motion or flashing	small
price	low cost	higher cost

Table A.2: Short summary of most notable differences between rolling- and global-shutter imagers

A short summary of the difference between rolling- and global- shutter imagers is shown in Table A.2

A.3 Optics

The second part of a camera is the optics or lens, used to manipulate reflected light rays from the environment, such that a proper image can be recorded at the imager. This section introduces the basic terminology and (simplified) mathematical relations of lens properties using the thin lens approximation.

One of the most well known equation in optics is the Gaussian (thin) Lens Formula:

$$\frac{1}{u} = \frac{1}{f} - \frac{1}{v} \tag{A.9}$$

It describes the relation of the distance u between the lens and an object placed in front of the lens, the focal point of the lens, f and the distance v at which the projection of the object is in focus.

Objects which are not located at a distance u are strictly speaking out of focus, yet objects close to u might appear in focus. The region $[u - \delta_n, u + \delta_f]$ is called the Depth-of-Field (DoF). Related to the DoF is the Circle-of-Confusion (CoC), which describes the maximum allowed blur for a spot to appear as it is focused. Controlling the DoF is straightforward: the focal point f determines the position of the DoF whereas the aperture controls the CoC and therefore the size of the DoF. A schematic overview is shown in Figure A.7.

The aperture is the diameter, d, of the lens opening through which light can pass. A smaller aperture results in a more focused light beam and therefore a smaller CoC. Using the f-number, $f/\#$ or N, the aperture can be expressed as a fraction of the focal point:

$$f/\# = N = \frac{f}{d} \tag{A.10}$$

While a smaller aperture increases the DoF and hence the overall sharpness of the image, it decreases the number of photons reaching the imager and therefore requires a longer shutter time to create an image with equal pixel intensities. As described in Chapter A.2.3, higher shutter times also increase the sensitiveness to motion blur.

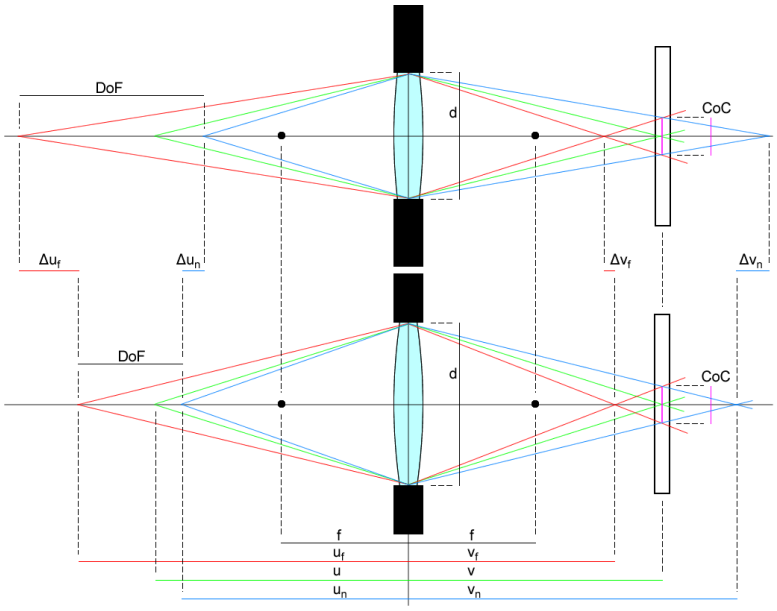


Figure A.7: Schematic overview of the relations of a simple lens between aperture (d), Circle of Confusion, Depth of Field and the Gaussian Lens approximation. As the top image has a smaller aperture, the depth of field is increasing, given the same geometrical constrains and CoC.

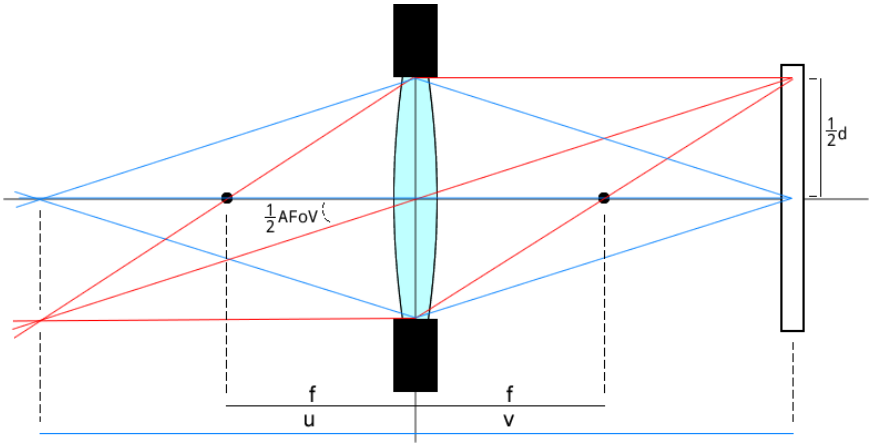


Figure A.8

In Figure A.7 a comparison is shown between two images with the same focal point f and CoC, but different aperture and therefore a different DoF.

The DoF is described by u_f and u_n, which describe the maximum and minimum (or hyperfocal) distance at which an object is projected with a blur smaller than the CoC, and is hence perceived as 'sharp'. From Figure A.7, u_f and u_n can be derived and expressed as:

$$u_n = \frac{vf}{v + Nc - f} \tag{A.11}$$

$$u_f = \frac{vf}{v + f - Nc} \tag{A.12}$$

$$\tag{A.13}$$

Or, when $v = f$, that is, the imager is placed at the focal distance, u_n simplifies to the following as $u_f = \infty$:

$$u_{n,v=f} = \frac{f^2}{Nc} \tag{A.14}$$

To conclude this section a notion has to be made on the Angular-Field-of-View (AFoV or FoV) of the lens. This is the subtended angle of the camera lens of the maximum view it can project on the imager. Figure A.8 shows a schematic overview. For the shown lens, assuming that the lens and imager are centrally aligned along the optical axis, the following equation for computing the FoV can be derived:

$$FoV = 2 * tan^{-1}\left(\frac{\frac{1}{2}d}{v}\right) = 2 * tan^{-1}\left(\frac{d}{2v}\right) \tag{A.15}$$

A.4 Image projection

While the previous chapters about pixels, imagers and thin lens approximations provide a basic understanding of a camera internals, it is not suitable to model the inaccuracies, distortions and complex interactions found in real-world setups. For this, more advanced models are used, as will be explained in Chapter 3.1.

This section, however, introduces several commonly used projections. The best, and perhaps widest, known projection is the *rectilinear projection*. All introduced math and insights from the previous sections are applicable to this projection. It is a projection in which straight lines from the real world appear as straight in the projection. The resulting image can be viewed in the same way as humans observe the world.

There are however also different projections possible such as fish-eye or wide-angle. Several attempts have been made to capture the complexities of the most common projection in simplified approximations [16, 34, 28]. The result is a set of five generic descriptions. Each assumes that a ray of light can be described with (θ_x, θ_y), representing the angle towards the optical axis of the lens in 3D space, projected to a 2D pixel location, described with the coordinate (x, y). For explaining the difference we assume that the ray of light moves from a 2D word coordinate to a 1D pixel position:

$$y = v * tan(\theta) \quad \text{rectilinear projection} \tag{A.16}$$

$$y = 2v * tan(\theta/2) \quad \text{stereographic projection} \tag{A.17}$$

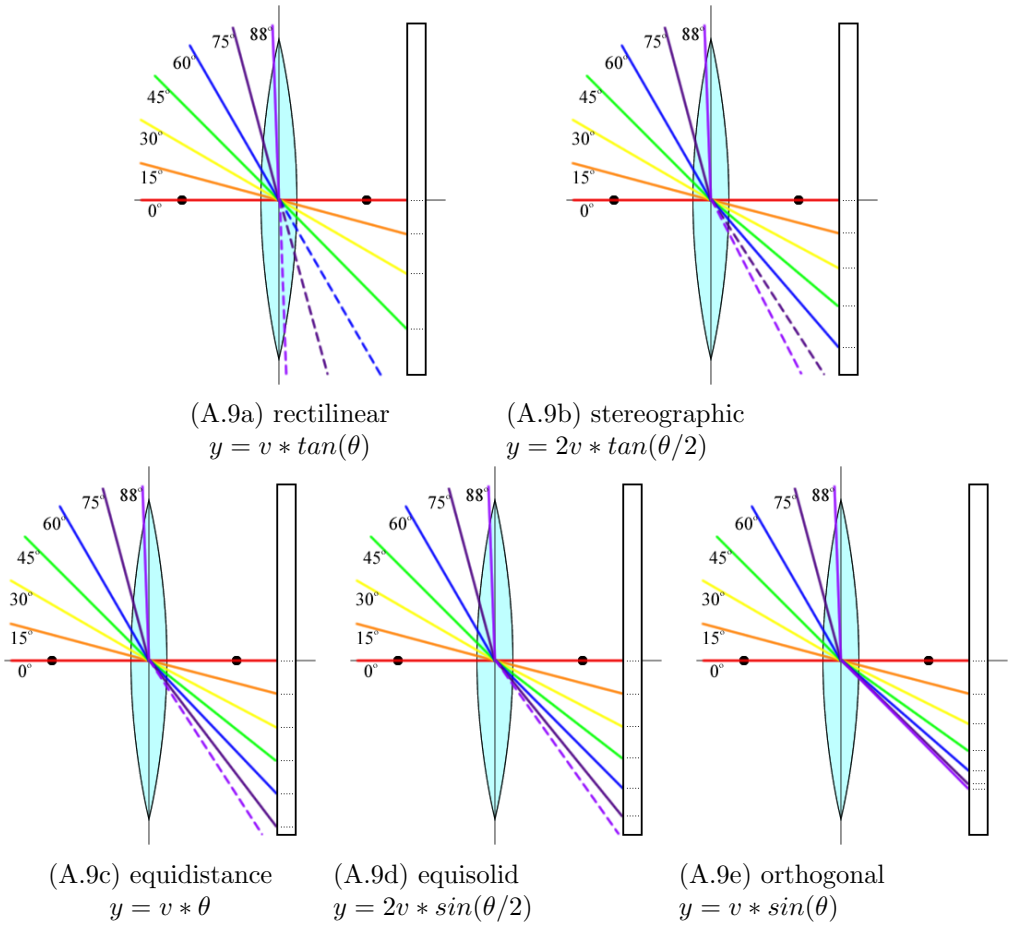
$$y = v * \theta \quad \text{equidistance projection} \tag{A.18}$$

$$y = 2v * sin(\theta/2) \quad \text{equisolid angle projection} \tag{A.19}$$

$$y = v * sin(\theta) \quad \text{orthogonal projection} \tag{A.20}$$

Where each projection produce a different result, as shown in Figure A.9.
In short, the difference between these projections are [43]:

- *Rectilinear*, $\mathcal{P}_{rect}$: straight lines appear straight on the projection.
- *Stereographic*, $\mathcal{P}_{stereo}$: crossing lines cross at the same angle in the projection, areas are however distorted (not 'equal area'). FoV commonly limited to 181 degrees.
- *Equidistance*, $\mathcal{P}_{eqdist}$: angle of ray is linear to the distance from the center of image. This projection is, in essence, the "ideal" fisheye. Widest recorded FoV is 220, but the FoV is typically around 182-185 degrees.
- *Equisolid*, $\mathcal{P}_{eqsolid}$: 'equal area' version of equidistance. Regions which have the same surface area in the real world appear to have the same surface area in the projection. Unlimited FoV (360), however, above 190 degrees, the image is heavily compressed and hence hard to use for image processing.
- *Orthogonal*, $\mathcal{P}_{ortho}$: Limited FoV (180), above 165 degrees the image is heavily compressed.



(A.9a) rectilinear
$y = v * tan(\theta)$

(A.9b) stereographic
$y = 2v * tan(\theta/2)$

(A.9c) equidistance
$y = v * \theta$

(A.9d) equisolid
$y = 2v * sin(\theta/2)$

(A.9e) orthogonal
$y = v * sin(\theta)$

Figure A.9: The five most common used projection models for modelling the effect of lenses and their effect on incoming light rays. Striped rays do not hit the imager.

Appendix B

Hardware Selection : Experiments and Results

This chapter shows the steps taken to select the hardware components for the prototype. It builds upon the models and ideas explained in Chapter 3 and Chapter A. First, an imager is selected, as it is the most limiting component. The imager is the least interchangeable and its properties such as pixel size, sensitiveness, exposure time and frame dimensions have all a large effect on projection, processing capabilities, and eye-safety. After the imager is chosen, the processing unit is determined, followed by several experiments to select the lens. Finally, a deduction is made on the safety of available laser modules after which an overview is given of the total estimated cost.

Before selecting the components, lets first recite the requirements:

1. Solid-state with a cost of $<$ \$30 each, when producing 1000 pieces.
2. Dimension should be within 10x3x4cm.
3. $>$ 2 meter detection range with a resolution $<$ 1%.
4. Planer, horizontal field-of-view of 120 degrees with a resolution of $<$ 1 degree.
5. $>$ 10Hz update rate. That is, 10 times a full 120-degree scan per second.
6. Eye-safe, Class 1 Laser product according to the NEN IEC 60825-1:2014 [24]
7. Fully embedded processing.

B.1 Imager and processing unit

The imager and processing unit determines a large part of the limitations of the system. This section analysis available components, their limitations and how they match with the stated requirements. A selection is made after which tests are performed to determine the throughput, which eventually will allow deduction of an eye-safe laser module.

conrad.nl	watterott.com	nl.farnell.com
robotshop.com (eu/fr)	digikey.com	nl.mouser.com
arducam.com		

Table B.1: Online stores from which the camera board is selected.

B.1.1 Imager

A vast amount of different imagers are nowadays on the market. For example, Mouser Electronics (mouser.com) alone, has at the moment of writing more than 750 different imagers available in its online catalog. As we are building a prototype, the complexity of designing and building a PCB on which an imager needs to be aligned is removed by only looking at off-the-shelve camera development boards. Since the finalized product should be able to be produced below $30, the set of potential camera boards is reduced by limiting the price to $70. Additionally, only boards with an M12 lens mount are selected as these allow flexibility in terms of adjusting the focus distance and allow to exchange or swap the lens. In Table B.1 the list of queried hardware stores is shown.

From these stores, a short list of potential boards has been selected, shown in Table B.2. Included in the overview are previously discussed camera parameters (Chapter A) as well as scaling and binning options, dynamic-range, and availability of a so-called strobe pin for (potentially) laser-camera synchronization.

Viewing Table B.2, several notions can be made:

- None of four matching imagers are monochrome and all are rolling shutter imagers.
- Several properties are unknown or not clearly defined and hence difficult to compare.
- The *OV5642* and *OV5647* have comparable properties, with the *OV5642* lacking the strobe, hence favouring the *OV5647*.
- The *OV5647* and *IMX219* boards have a strobe-pin. This pin is specially designed to drive an external (LED) flash and therefore can be used to drive the laser.
- The *OV7670* is used in [17, 38], which shows in Chapter 2.1 that it might meet the minimal requirements.
- The *IMX219* is more recent, has higher resolution, a strobe pin and is able to run at higher speeds, hence is favored over the *OV7670*.
- Only the *IMX219* is still in production. The others are End-Of-Line (EOL).

Based on these observations the IMX219 board has been selected as the most suitable option.

Imager release EOL	price url source	DR SNR (dB)	reso@fps	scaling Bin. H/V fps	pixel (μm)	Sensitivity Dark Cur. ratio	Well (e^-)	sync[13]
OV5642 09‘08 09‘14[15]	$32 [1] [2,3]	68 36	2592x1944@15 1920x1080@30 1280x960@45 1280x720@60 640x480@90 320x280@120	full + crop 1-2/1-2 ?	1.4x1.4	680 $\frac{mV}{lux*sec}$ 15 $\frac{680}{15} \sim 45.3$	?	(strobe)
OV5647 02‘10 07‘15[16]	$32 [4,5] [6,7]	68 36	2592x1944@15 1920x1080@30 1280x960@45 1280x720@60 640x480@90 320x280@120	full + crop 1-2/1-2 $\frac{PLL}{2048*(v+24)*t_p}$	1.4x1.4	680 $\frac{mV}{lux*sec}$ $16mV@60C$ $\frac{680}{16} \sim 42.5$	4.3k	strobe
OV7670 02‘06 01‘14[17]	6 € [8] [9,10]	? 38	640x480@30 320x240@30 160x120@30 352x288@30 176x144@30	full + crop ? $\frac{PCLK}{510*784*t_p}$	3.6x3.6	1100 $\frac{mV}{lux*sec}$ $12mV@60C$ $\frac{1100}{12} \sim 91.6$	17k	(strobe)
IMX219 04‘14 -	$25 – $65 [11] [12]	? ?	3240x2464@15 1920x1080@30 1280x720@60 640x480@90	full + crop 1-2-4/1-2-4 $\frac{2*PCLK}{H*V}$	1.12x1.12	203 LSB [14] 0.5 LSB@60C $203/0.5 \sim 406$	?	strobe

Table B.2: Overview of parameters and errors in comparable systems

[1] http://www.robotshop.com/eu/en/arducam-5-mp-camera-module-ov5642-cs-mount-lens.html

[2] http://www.uctronics.com/download/cam_module/OV5642DS.pdf

[3] http://www.ovt.com/uploads/parts/OV5642_PB_v1_1_WEB.pdf

[4] http://www.robotshop.com/eu/en/arducam-5mp-1080p-ov5647

[5] http://www.watterott.com/de/Raspberry-Pi-Camera-Board-/w-CS-mount-Lens

[6] http://www.ovt.com/uploads/parts/OV5647.pdf

[7] http://cdn.sparkfun.com/datasheets/Dev/RaspberryPi/ov5647_full.pdf

[8] http://www.robotshop.com/eu/en/arducam-cmos-ov7670-camera-module.html

[9] http://www.voti.nl/docs/OV7670.pdf

[10] http://aitendo3.sakura.ne.jp/aitendo_data/product_img/camera/OV7690/

[11] Stated in email-exchange with http://arducam.com. Including a $25 fee for the Pi Foundation

[12] Datasheet retrieved from sony after request [4]

[13] Laser synchronisation options. ‘(...)’ indicate ‘datasheet only’.

[14] Expressed by the resolution of the 10-bit ADC.

[15] http://www.ovt.com/download_document.php?type=document&DID=97

[16] http://www.ovt.com/download_document.php?type=document&DID=122

[17] http://www.ovt.com/download_document.php?type=document&DID=95

B.1.2 Processing unit

Just as with imagers, there exists a large variety of processing boards. Even the amount of available boards below a $10 price point is enormous. However, when factoring in the use of the IMX219, the Raspberry Pi Zero and its wireless variant the Raspberry Pi Zero W show promising specifications. Their prices are EUR5 and EUR11 respectively and there form factor is around 65x30x5mm, fitting our requirements. Furthermore, the IMX219 is the official imager for the Raspberry Pi boards, hence from a development point of view, there are already development tools and documentation available in addition to a fully configured ISP.

The Raspberry Pi Zero (W) is built around the BCM2835 SoC [1] and holds the following specifications:

- A Broadcom BCM2835 SoC
  - 1GHz ARM11 (ARM1176JZF-S)
  - VideoCore IV GPU running the Real-Time Operating System ThreadX and including an ISP.
- 512MB of LPDDR2 SDRAM stacked ontop of the BCM2835.
- 1x A mini-HDMI socket for 1080p60 video output
- 2x Micro-USB sockets for data and power
- 40-pin GPIO header
- 65mm x 30mm x 5mm

It is suspected that when both CPU and GPU are used, the Raspberry Pi Zero should be able to meet our requirements and is therefore selected.

B.1.3 Interfacing

Having selected both imager and processing unit, a closer look can be taken in how these components are connected and how frames are captured. In Figure B.1 a schematic overview of some of the inner components of the BCM2835 are shown and their connection with the imager. A better understanding will yield into better estimates of the limits of the system.

To capture a frame, a user specifies several camera parameters (e.g. resolution, framerate, etc) in the CPU domain. This is followed by the following (simplified) sequence of steps:

1. The user-defined camera parameters values are pushed via several abstraction layers such as MMAL (MultiMedia Abstraction Layer [2]) and VCHI (VideoCore Host Interface [3]) to the GPU.

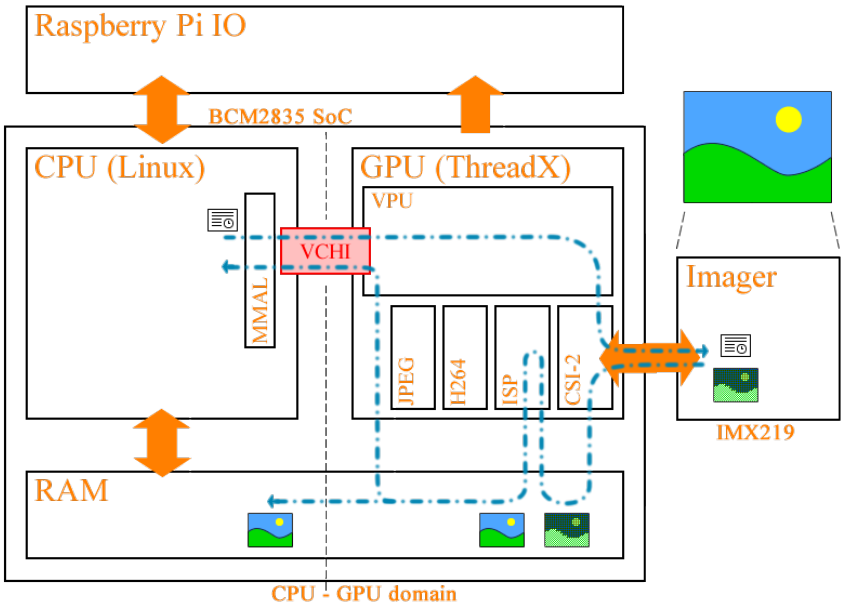


Figure B.1: Overview of how the CPU, GPU and camera communicate [27, 1, 3].

2. The GPU communicates the settings to the imager via a CSI-2 interface (also called 'Unicam').

3. Depending on the settings, the imager either:

   (a) produces a single frame ('capture' mode)

   (b) produces a stream of frames at a given fps ('video' mode)

4. The GPU receives the raw-Bayer frames from the imager, adds a timestamp and stores them in a buffer.

5. Frames are then retrieved from the buffer and pushed through a dedicated 14-stage ISP [27] which applies operation such as demosaicing, image resizing, white balancing and several others.

6. After being pushed through the ISP, the frames are via DMA copied to the CPU domain for further (user-)processing. In addition, a callback is signaled which notifies the CPU that a frame is ready.

Mode	Frame size (CSI-2)	Framesize (effective)	Binning	Crop[1] (pixels)	Readout[1] (ns / line)	Capturing (ms)	Framerate (Hz)
0	automatic						
1	1920x1080	1920x1080	-	680 / 692	18904	20.4	48.98
2	3280x2464	3280x2464	-	- / -	18904	46.6	21.47
3	3280x2464	3280x2464	-	- / -	18904	46.6	21.47
4	1640x1232	3280x2464	2x2	- / -	18904	23.3	42.94
5	1640x922	3280x1844	2x2	- / 310	18904	17.4	57.37
6	1280x720	2560x1440	2x2	360 / 512	19517	14.1	71.16
7	640x480	1280x480	2x2	1000 / 752	19517	9.36	106.74

Table B.3: Sensor modes of the Raspberry Pi firmware for the IMX219. The first 6 columns are documented limits, the final two are computed based on these values. Column three states the effective resolution (or pixel area) which is used by the imager. Cropping is written as 'left&right / top&bottom'. The column 'Readout' states the number of nanoseconds which the imager needs to read a single line from the photosensitive area. Based on this value the capture time of a single frame is computed from which the maximum theoretical framerate is deduced.

[1] `https://www.raspberrypi.org/forums/viewtopic.php?t=177046`

B.1.4 Modi and throughput

Taking a look at the available user-settings for the camera shows that the Raspberry Pi Foundation has predefined 8 different capture-modi in the GPU firmware. So instead of tweaking and updating firmware and register settings of the imager, we can use predefined resolutions and update rates, which will set the proper register automatically. An overview of the specification for each mode is shown in Table B.3.

To validate the estimated specifications and throughput of the pipeline, stability-experiments have been performed. These experiments try to force a framerate of 200Hz and log the timestamp at which frames are received at the CPU. Results are displayed in Table B.4. Note that mode 0 is not tested as this mode selects mode 1-7 automatically based on the user settings.

When looking at Table B.4, most interestingly, not all measured framerates match with the documented or the theorized limits. It is suspected that these values are due to inaccuracies of the firmware. Unfortunately, the firmware is closed source, so the exact cause cannot be determined.

In addition to the maximum framerate, the stability of the minimum required framerate is tested. The same settings as for Table B.4 are applied, except the frequency is set to 15Hz. Results are shown in Table B.5. A graphical overview of the computed duty cycle of the total exposure time at the minimum framerate is shown in Figure B.2.

The maximum allowable power of the laser is inversely related to the time it is powered on (Chapter 3.3), hence modes with a low duty cycle are preferable. Table B.3 - B.5 and Figure B.2 show that mode 5 and 7 have the lowest duty cycle, while matching the requirements. In Table B.3 it is stated that mode 5 utilises the full imager width, while

	Docs		Theory	640x480		
Mode	min	max	max	$\mu \pm \sigma$	max	Mpix/s
1	0.1	30	48.98	30.1 ± 0	30.1	62.4
2	0.1	15	21.47	18.7 ± 4.6	21.4	151.1
3	0.1	15	21.47	18.7 ± 4.6	21.4	151.1
4	0.1	40	42.94	60.0 ± 0	60.0	121.2
5	0.1	40	57.37	60.0 ± 0	60.0	90.7
6	40	90	71.16	109.9 ± 30.4	131.5	101.28
7	40	90	106.74	120.5 ± 0	120.6	37.0

Table B.4: Maximum framerate limits. Minimum and maximum frequency according to the documentation, theory (Table B.3) and with the ISP set to resize to 640x480 pixels. The last column shows the number of megapixels communicated per second via CSI-2. It is based on the average fps and framesize of the produced frame from the imager. Each mode is ran for 1000 frames. The theorised maximum[1] throughput for the ISP is stated to be 200Mpix/s, while it is typical between 150-150 Mpix/s

. [1] `https://www.raspberrypi.org/forums/viewtopic.php?t=56503`

	Docs		640x480			
Mode	min	max	$\mu \pm \sigma$	min	Mpix/s	Dutycycle (%)
1	0.1	30	15.0 ± 0	15.0	31.1	30.6
2	0.1	15	15.0 ± 0	15.0	121.2	69.9
3	0.1	15	15.0 ± 0	15.0	121.2	69.9
4	0.1	40	15.0 ± 0	15.0	30.3	34.9
5	0.1	40	15.0 ± 0	15.0	22.7	26.1
6	40	90	40.2 ± 0	40.2	37.0	56,5
7	40	90	40.2 ± 0	40.2	12.3	37.7

Table B.5: Test results when the frameset is set to the required minimum of 15Hz. Each mode is ran for 1000 frames. The column 'Dutycycle' represents percentage in which the imager is exposing its pixels at the minimum measured frequency during 1 second, which is determined by multiplying the framerate with the minimum capturetime of the mode.

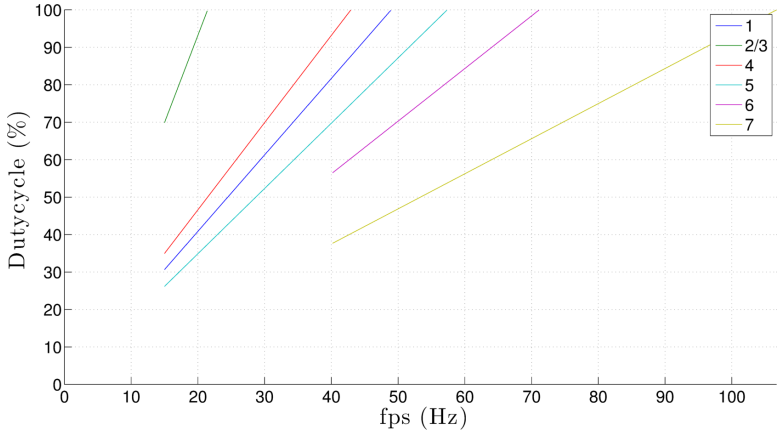


Figure B.2: Dutycyle of the exposure versus the frame rate of differen modi. Computations are based on the minimal required (measured) framerates and the theorised capture time.

mode 7 is cropped. As more pixels result in an improved angular resolution, mode 5 is selected for the prototype.

B.2 Lens

After determining the imager and processing unit, this section presents the selection procedure of the lens.

B.2.1 Preselection

The requirements state that a minimal FoV of 120 degrees is required. In addition, the lens should be suitable for at least 1/4" imagers (diameter of the IMX219) and should have M12 threading (Chapter B.1). In Table B.6 an overview of available lenses is given. This overview is constructed by filtering matching lenses from `http://arducam.com`, which supplies the selected camera board.

Using the generalized projections presented in Chapter A.4, the maximum diameter of a projected image at the focal distance was computed. The results are shown in Table B.7. As can be observed, several models state that an image will be produced, smaller than the diagonal of the IMX219 (4.59mm), resulting in black and unusable edges (vignetting) of the captured frame. In order to select a proper lens, the inverse computation is also made: computing the maximum FoV by using the diameter of the IMX219. These results are shown in Table B.8.

By filtering out vignetted lenses and preferring larger apertures, a small set of potential lenses can be selected. The resulting selection is shown in Table B.9. Experiments will show which of these lenses is the most proper match.

Lens	AFoV (dia)	Type	f	$f/\#$	Distortion	Aperture
LS-36021	120^o	1/4"	2.8	2.2	$< -45\%$	1.27 mm
LS-40207	120^o	1/4"	2.8	2	$< -35\%$	1.4 mm
LS-1820	150^o	1/4"	1.8	2.4	$< -83\%$	0.75 mm
LS-50125	170^o	1/4"	1.6	2	$< -20\%$	0.80 mm
LS-40180	206^o	1/4"	1.05	2	$< -81\%$	0.53 mm
LS-40146	204^o	1/3.2"	1.52	2.4	$< -97\%$	0.63 mm
LS-30207	120^o	1/3"	2.8	2	$< -35\%$	1.4 mm
LS-27225	145^o	1/3"	2.1	2	$< -68\%$	1.05 mm
LS-30188	170^o	1/3"	2.25	2.3	$< -81\%$	0.98 mm
LS-4014	140^o	1/2.5"	3	2	$< -30\%$	1.5 mm
LS-20150	160^o	1/2.5"	2.8	2.8	$< 39\%$	1.00 mm
LS-20233	170^o	1/2.5"	2.8	2.8	-	1.00 mm
LS-25180	185^o	1/2.5"	1.6	2	$< -5.7\%$	0.8 mm
LS-81600	155^o	1/2.3"	2.8	2.8	$< 55.6\%$	1.00 mm

Table B.6: Overview of potential suitable lenses from `arducam.com`

Lens	$\mathcal{P}_{stereo}$	$\mathcal{P}_{eqdist}$	$\mathcal{P}_{eqsolid}$	$\mathcal{P}_{ortho}$
LS-36021	6.47	5.86	5.60	4.85
LS-40207	6.47	5.86	5.60	4.85
LS-1820	5.52	4.71	4.38 **	3.48 **
LS-50125	5.86	4.75 **	4.32 **	3.19 **
LS-40180	5.28	3.78	3.29 **	2.05 **
LS-40146	7.51	5.41	4.73	2.97 **
LS-30207	6.47	5.86	5.60	4.85
LS-27225	6.16	5.31	4.97	4.01 **
LS-30188	8.25	6.68	6.08	4.48 **
LS-4014	8.40	7.33	6.88	5.64
LS-20150	9.40	7.82	7.20	5.51
LS-20233	10.26	8.31	7.57	5.58
LS-25180	6.69	5.17	4.62	3.20 **
LS-81600	8.99	7.57	7.01	5.47

Table B.7: Maximum diameter of image given de models and lens parameters, with $f = v$. Values with ** indicate vignetting.

Lens	$\mathcal{P}^{-1}_{stereo}$	$\mathcal{P}^{-1}_{eqdist}$	$\mathcal{P}^{-1}_{eqsolid}$	$\mathcal{P}^{-1}_{ortho}$
LS-36021	89.14	93.92	96.77	110.10
LS-40207	89.14	93.92	96.77	110.10
LS-1820	130.07	146.10	150 **	150 **
LS-50125	142.59	164.37	170 **	170 **
LS-40180	190.16	206 **	206 **	206 **
LS-40146	148.20	173.02	196.08	204 **
LS-30207	89.14	93.92	96.77	110.10
LS-27225	114.61	125.23	132.49	145 **
LS-30188	108.09	116.88	122.66	170 **
LS-4014	83.73	87.66	89.95	99.81
LS-20150	89.14	93.92	96.77	110.10
LS-20233	89.14	93.92	96.77	110.10
LS-25180	142.59	164.37	183.29	185 **
LS-81600	89.14	93.92	96.77	110.10

Table B.8: Maximum diagonal AFoV of IMX219 given de models and lens parameters, with $f = v$. Values with ** indicate an FoV which matches the requirements.

Lens	$\mathcal{P}_{stereo}$	$\mathcal{P}_{eqdist}$	$\mathcal{P}_{eqsolid}$	$\mathcal{P}_{ortho}$	f	Aperture	Price
LS-1820	130.07	146.10	150	150	1.8	0.75 mm	$8
LS-50125	142.59	164.37	170	170	1.6	0.80 mm	$8
LS-25180	142.59	164.37	183.29	185	1.6	0.8 mm	$25

Table B.9: Remaining lenses with sufficient diameter and aperture

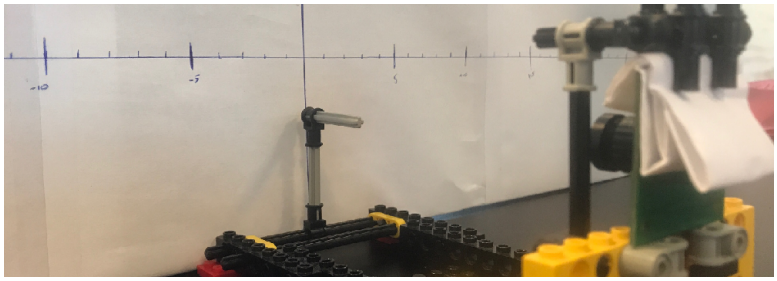
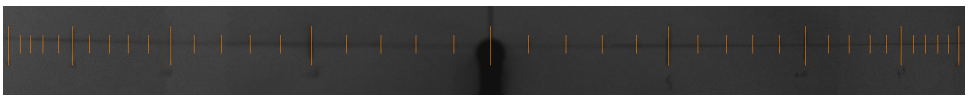
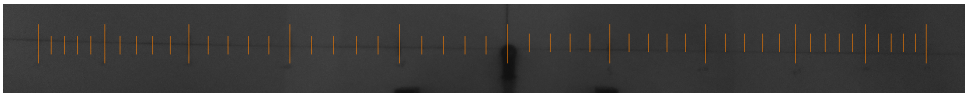


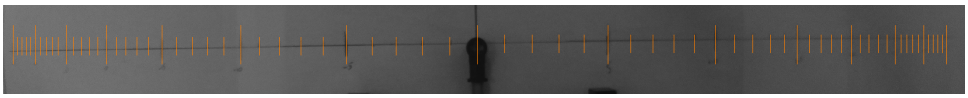
Figure B.3: Picture of the setup to measure the FoV of the lenses in Table B.9. A ruler measuring $[-35cm, 35cm]$ has been drawn to a wall (left) and the camera (right) is placed at a distance of $\sim 16.5cm$.



(B.4a) LS-1820



(B.4b) LS-50125



(B.4c) LS-25180

Figure B.4: Captured frames with manually placed markers at each cm for the selected lenses.

B.2.2 Experiments

Determining the FoV of the lenses is done with the setup shown in Figure B.3. The camera is facing a ruler of $[-35, +35]$cm at a distance of 16 cm. After taking a snapshot with the camera the maximum observable value of the ruler is determined manually. The angular accuracy of the lens is estimated by including the resolution of the imager. For these tests, mode 2 of the imager is used: this mode is the most generic and from its frames, we can deduce the effects of mode 3-7, since these are cropped and binned versions of this mode.

The captured frames for all lenses described in Table B.9 are shown in Figure B.4.

For each captured frame markers are placed at the image to identify each centimeter. As the marker is placed within a frame, its horizontal position, m_x, can be described with:

$$m_x \in [0, 3280] \tag{B.1}$$

Polynomial	2	3	4	5	6	7	8	9
LS-1820	41.4	41.2	40.9	40.9	41.0	40.9	40.8	40.8
LS-50125	56.4	56.7	56.4	56.4	55.9	55.8	56.0	55.9
LS-25180	89.6	89.1	76.2	76.2	76.5	76.5	76.1	76.1

Table B.10: Maximum observable distance (cm) per polynomial fit, as computed by Equation (B.3). The selected mode is mode 2, utilising the full frame width.

Mode	1	2/3 /4/5	6	7
Crop	$[680, 2600]$	$[0, 3280]$	$[360, 2920]$	$[1000, 2280]$
LS-1820	17.1cm / 54.7^o	41.0cm / 102.3^o	25.4cm / 75.3^o	10.6cm / 35.6^o
LS-50125	27.5cm / 79.5^o	55.9cm / 118.9^o	39.2cm / 99.9^o	17.7cm / 56.5^o
LS-25180	25.0cm / 74.3^o	76.5cm / 133.3^o	40.1cm / 101.2^o	14.9cm / 48.7^o

Table B.11: Maximum observable distance (cm) for each mode and the resulting FoV using a 6-th order polynomial fit.

To compare the selected lenses, the derivative of the set of markers for each lens is computed:

$$\partial m = \frac{dm_x}{dx} \tag{B.2}$$

As ∂m describes the horizontal number of pixels per cm per pixel position, its inverse can be used to estimate the maximum observed horizontal distance, d, of the frame:

$$d = \sum_{x=s}^{e} \frac{1}{\partial m_x} \tag{B.3}$$

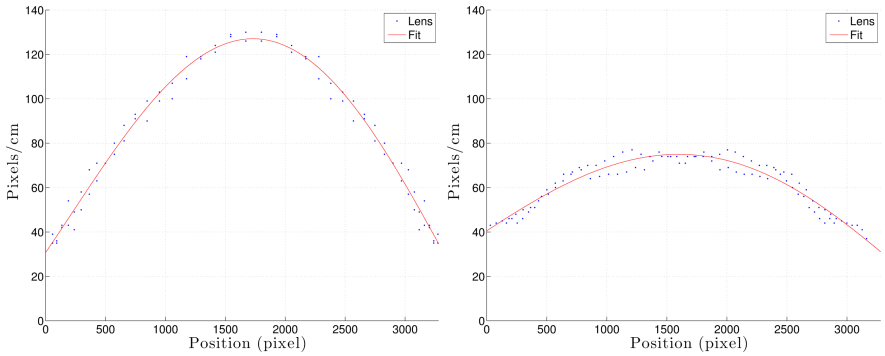
In this equation $[s, e]$ is the boundary of the cropped values for the selected mode. In case of mode 2, it equals $[s, e] = [0, 3280]$.

Having determined the maximum observable distance, the FoV is determined by:

$$FoV = \frac{180}{\pi} * tan^{-1}\left(\frac{.5d}{16.5}\right) \tag{B.4}$$

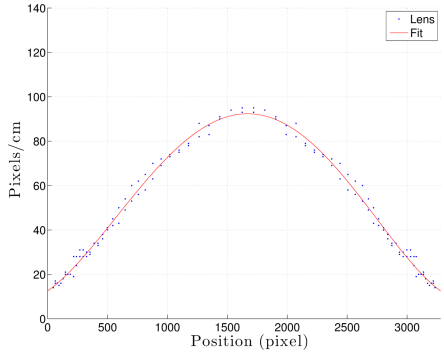
By applying a polynomial fit to the captured set of marker positions, an estimate d can be computed. An overview of the 2^{nd} - 9^{th} order polynomial fit for each lens is shown in Table B.10. From the table it can be estimated that a 6^{th} order polynomial is sufficient to compute the estimations, which is in line with the estimates of the radial distortions in the camera model (Chapter 3.1). Using the fit, estimations for ∂m, FoV and angular resolution are computed. Results are displayed in Figure B.5, Table B.11 and Figure B.6 respectively.

The requirements state that a minimum resolution of 0.9 degrees should be achieved. Looking at Figure B.6, the estimates for all lenses are below this value. Moreover,



(B.5a) LS-1820

(B.5b) LS-50125



(B.5c) LS-25180

Figure B.5: 6th order polynomial fit for ∂m for each lens. Note that the sample data for each lens is used in both normal and flipped direction ($x = [0, 3280]$ and $x = [3280, 0]$) as the manual annotation is inaccurate and asymmetric while the lenses should result in a symmetric projection.

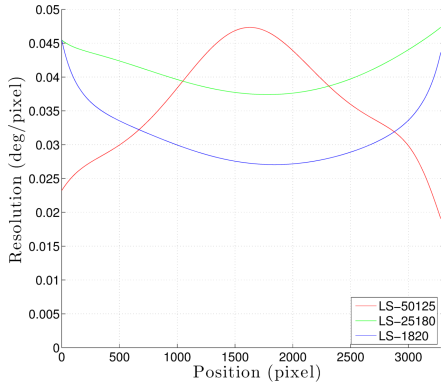


Figure B.6: Angular resolution of the lenses for mode 2. It is computed by applying the cosine-rule on the 6th-order polynomial fit. Note that when mode 4-7 is used, the resolution doubles due too the 2-binning mechanism.

when 2-binning is applied (mode 4-7) the lowest resolution is around 0.95^o for LS-50125, whereas LS-25180 and LS-1280 stay below 0.9^o, showing that all lenses are within reason of the angular-resolution requirement.

A different requirement is that the lens should have a 120^o FoV. Table B.11 shows that LS-1820 does not match this requirement. In mode 2/3/4/5 the LS-50125 has an estimated FoV of 118.9^o, which is only off by 1.1^o and the LS-25180 has an estimated FoV of 133^o. By factoring in the price of the lenses LS-50125 and LS-25180 ($8 versus $25), LS-50125 is favorable: while it approximates the minimum requirements, its cost is a factor of 3 times lower when compared to LS-25180.

B.3 Laser

The final hardware component to be determined is the line-laser. It should have a field-of-view (or fanning angle) of 120^o, while having an output power just below the eye-safety limit. From the IEC60825 safety standard [24], we can deduce that a higher wavelength, λ, produces less damage to the human eye than a lower wavelength. Unfortunately, using a CMOS imager, the QE of the pixels drop with an increase of λ. Therefore a wavelength of 808nm is chosen.

In Figure B.7 the NOHD values are shown for a laser with $\lambda = 808$nm, with a FoV of 120 degrees and a pulse frequency of 5, 10, 15 or 30Hz at a given duty cycle.

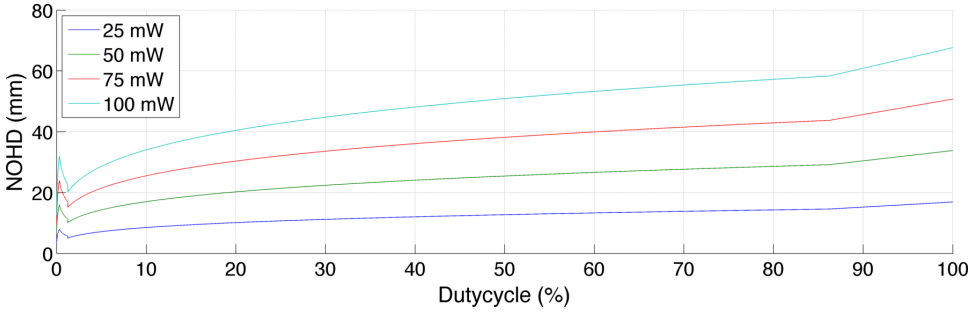
As expected, with an increase of the duty cycle or power, the distance for safe operation increases. The provided frequencies do not seem to have such a large effect.

To be a Class 1 product, we need to ensure that a user cannot view the laser within the NOHD a limit of 10mm is selected. To determine the maximum laser power, we need to determine the pulse width and hence duty cycle of the laser. As shown in the previous sections, the imager has an exposure time of 17.4ms when mode 5 is used (Table B.3). Assuming that the laser pulse equals the exposure time of the imager, we can deduce that only a 25mW laser falls within the set NOHD.

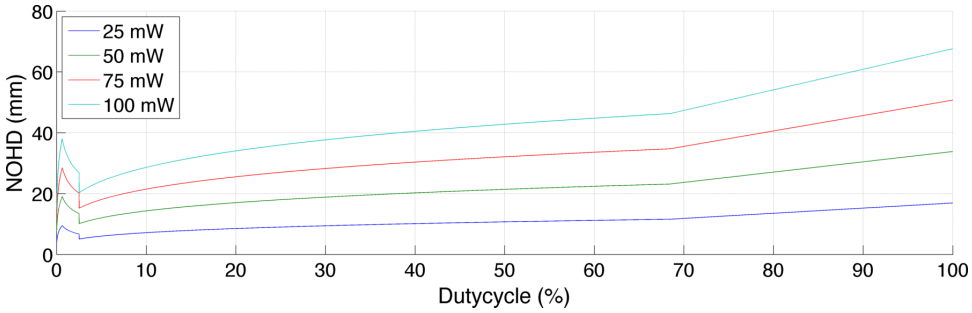
B.4 Overview

Having selected all hardware components, an overview can be created of all costs involved. The costs for the prototype (Chapter 4) and an estimation for producing 1000 units is shown in Table B.12.

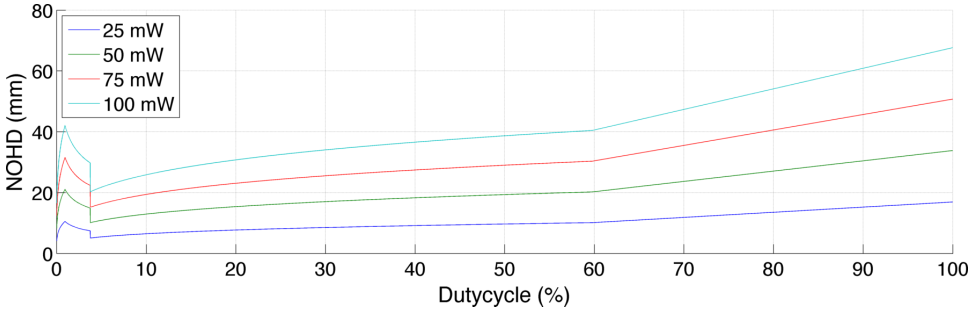
While the cost of 1kpcs, is above the required amount ($65 versus $30), it should be noted that this price includes licensing fees, is not based on hardware optimisations or selection of parts with the lowest cost and does not contain any negotiations. Hence, there is still a lot of room for reduction, indicating that the requirement eventually can be met.



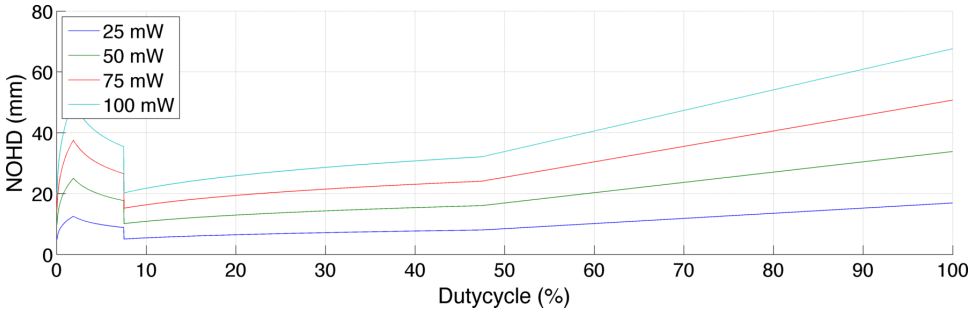
(B.7a) f=5Hz



(B.7b) f=10Hz



(B.7c) f=15Hz



(B.7d) f=30Hz

Figure B.7: NOHD versus dutycyle at different frequencies. For each setup yields that: $\lambda = 808$nm, $d = 1$mm, $T = 100$s, $fov = 120$ degrees and $y = 0.4$cm. The duration of a pulse is limited to the domain $[5 * 10^{-6}, 0.25]$s

Part	price @ 1pcs	est. price @ 1kpcs
Lens (LS 50125)	$8	$5.00
Camera board (M12, IMX219)	$65	$35.00
Bandpassfilter, 808nm ± 25nm	$10	$5.00
Raspberry Pi Zero W	$9.60	$9.60
Raspberry Pi Zero Camera Ribbon	$4.00	$1.00
2x20 female header	$1.50	-
16Gb Class 4 micro SDCard	$9	$3.00
5V Power Supply	$5	$1.00
25mW 808nm 120deg Lase Module	$12.5	$5.00
Custom laser driver	$5	$1.00
3D printed mount + screws	$8	$2.00
Total	$125.6	$63.00

Table B.12: Cost of a single module and estimated cost of 1kpcs. The price of the camera board is including a $25 licensing fee for the Pi Foundation as a custom board is used. Excluding the licensing fee, a single unit has a cost of $\sim$ $100 and an estimated cost $\sim$ $38 when producing 1000 pieces. Using a Raspberry Pi Zero will decreases the unit cost with an additional $5.

Appendix C

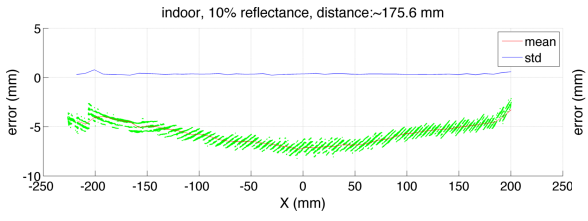
Measurements, Results and Data

This chapter list several measurements and test results. Based on these results, deductions and approximations have been made, for which Chapter 6 will give some more details.

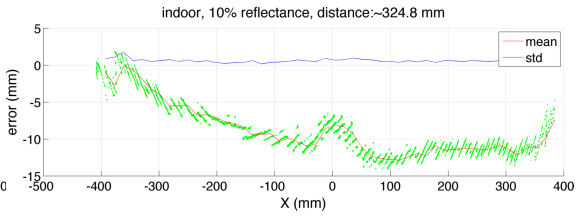
In the first two sections, the measurement error per measurement-set in an indoor setup with a 10% and 90% reflective wall is shown. The error is the difference between the distance estimations of the Hokuyo UTM-30LX [13] and de prototype, dubbed LaserPi. Computation of the average and the standard deviation is done by combining the measurements into bins, where each bin is determined by dividing the x-axis into 50 equally spaced areas.

Additionally, the angular resolution at different distances is shown, followed by several GPU traces, measuring the load of the VPU.

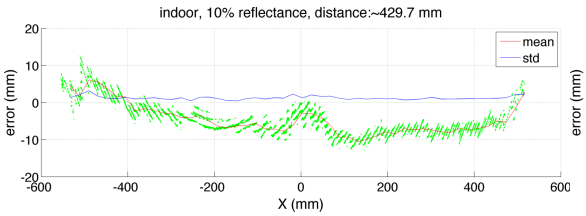
C.1 Measurement error: indoor, 10% reflection



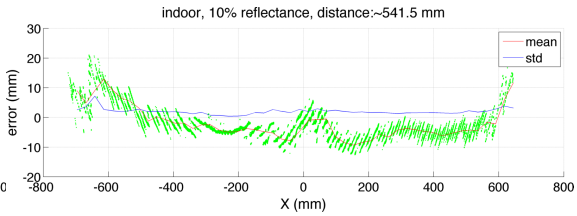
(C.1a)



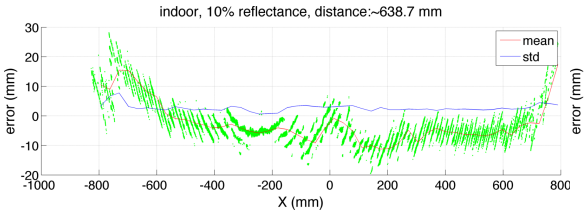
(C.1b)



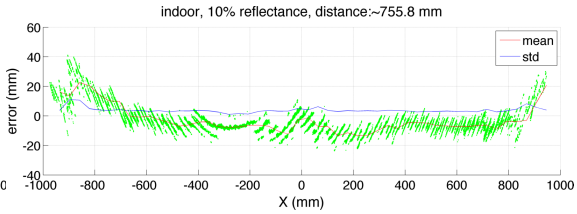
(C.1c)



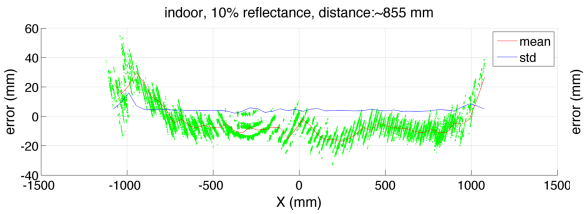
(C.1d)



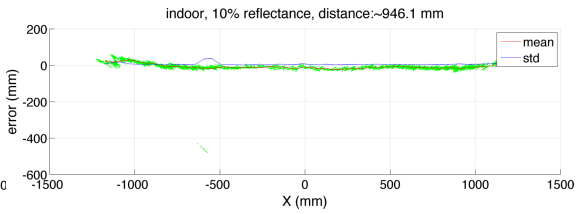
(C.1e)



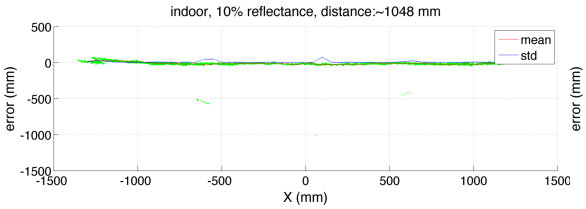
(C.1f)



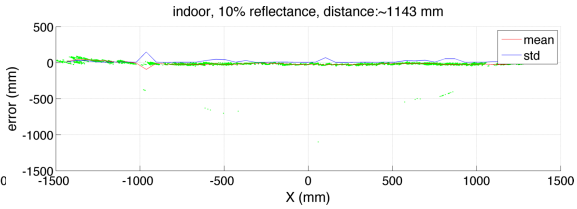
(C.1g)



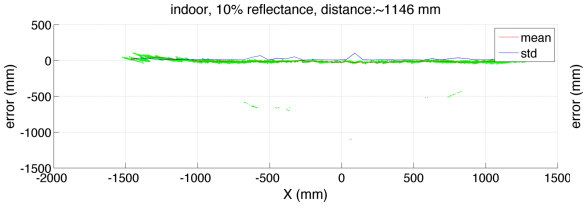
(C.1h)



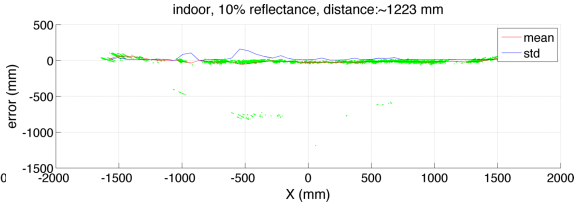
(C.1i)



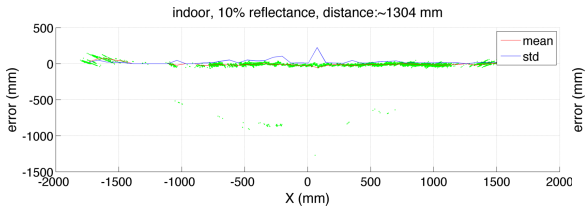
(C.1j)



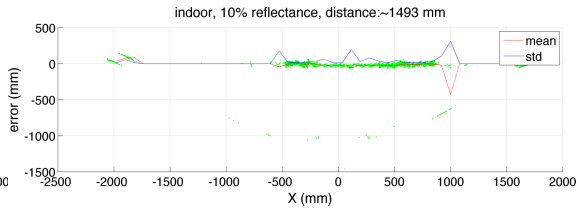
(C.1k)



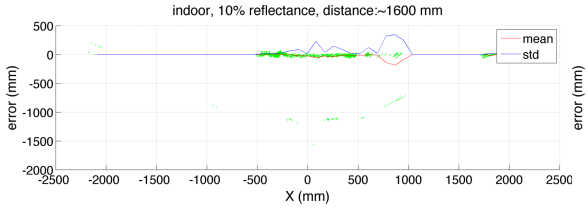
(C.1l)



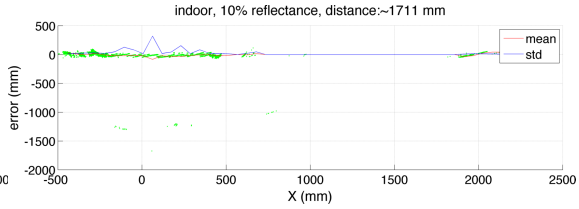
(C.1m)



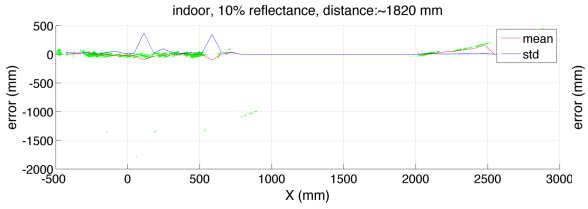
(C.1n)



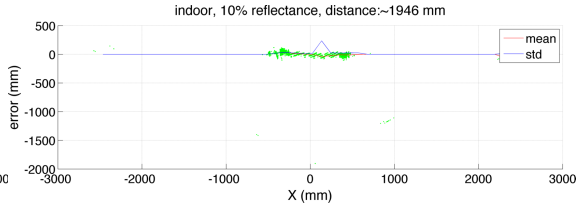
(C.1o)



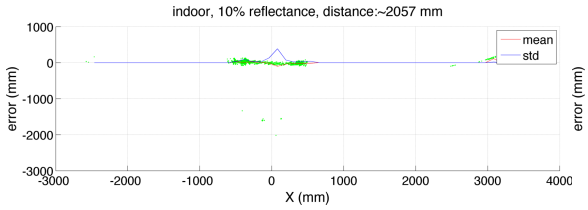
(C.1p)



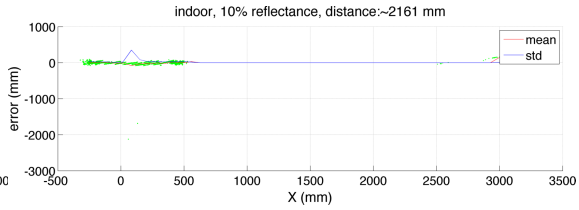
(C.1q)



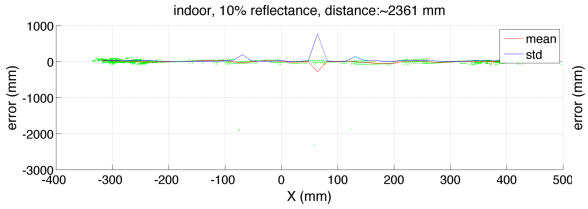
(C.1r)



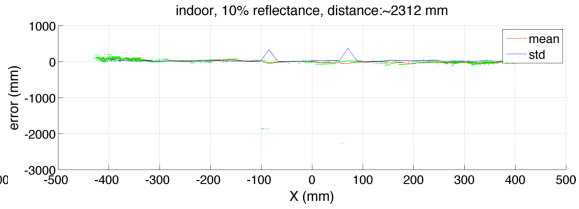
(C.1s)



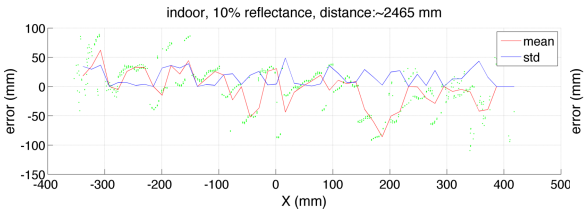
(C.1t)



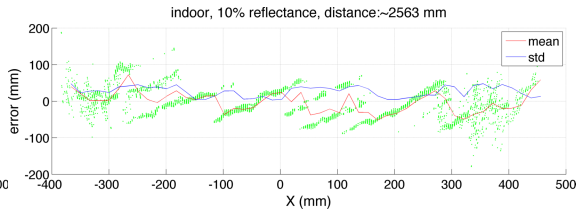
(C.1u)



(C.1v)

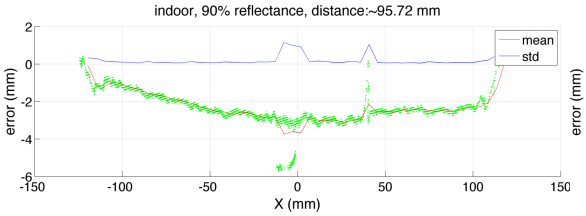


(C.1w)

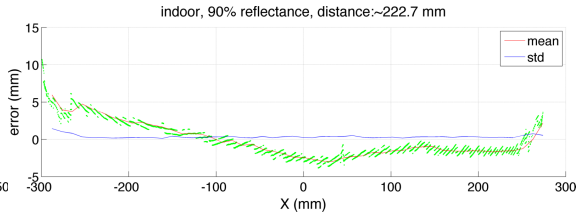


(C.1x)

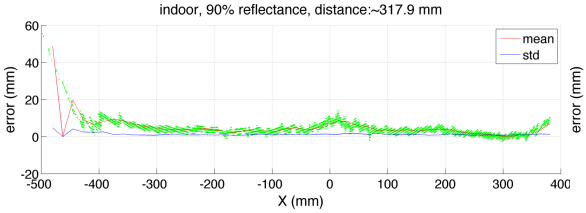
C.2 Measurement error: indoor, 90% reflection



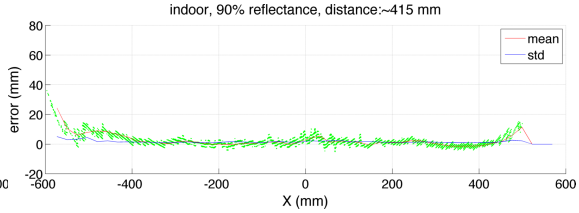
(C.2a)



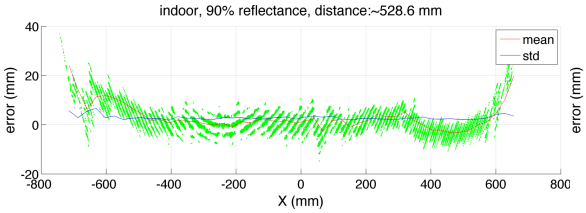
(C.2b)



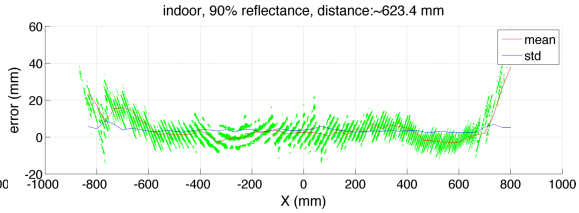
(C.2c)



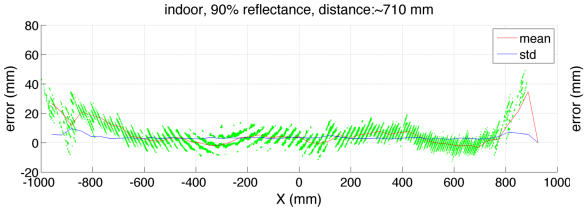
(C.2d)



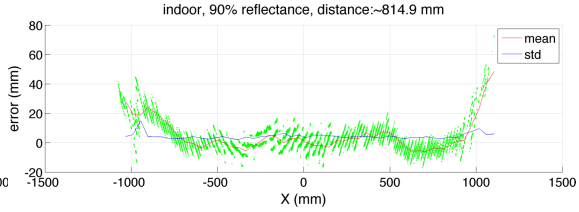
(C.2e)



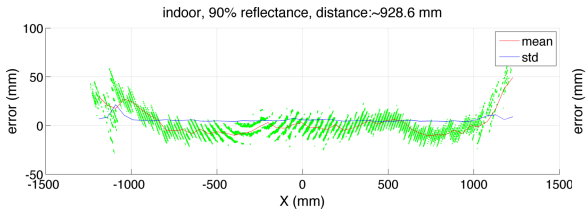
(C.2f)



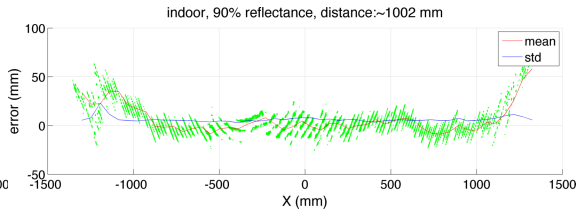
(C.2g)



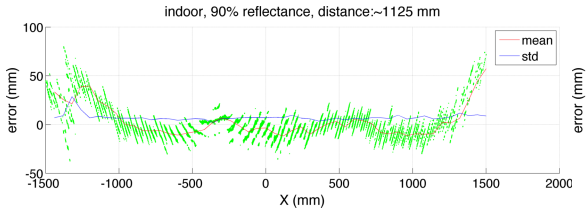
(C.2h)



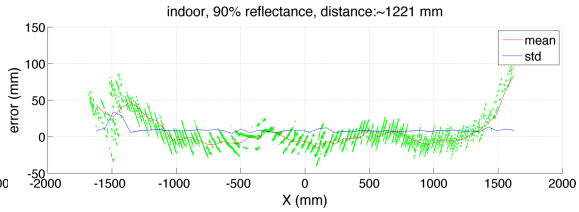
(C.2i)



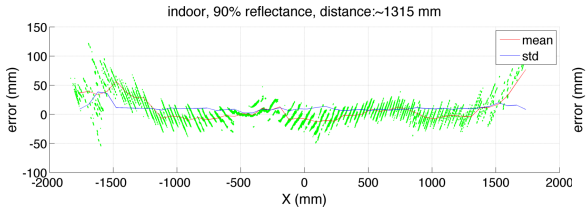
(C.2j)



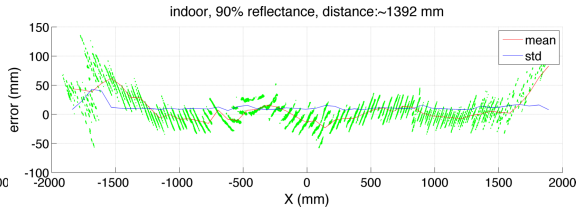
(C.2k)



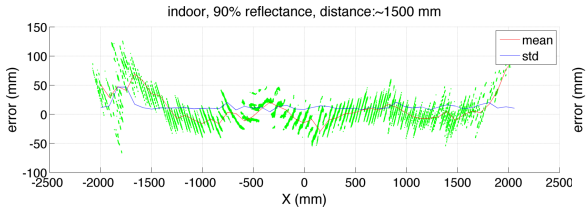
(C.2l)



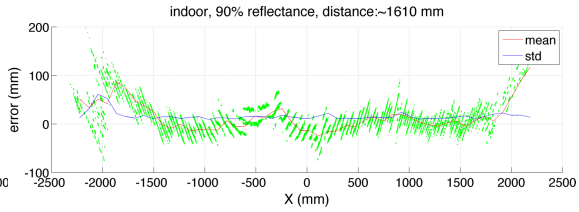
(C.2m)



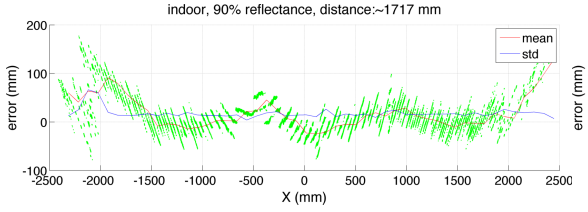
(C.2n)



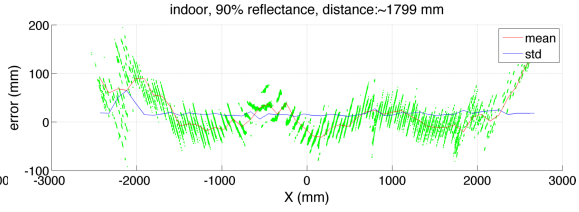
(C.2o)



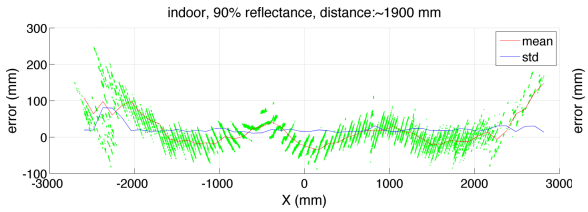
(C.2p)



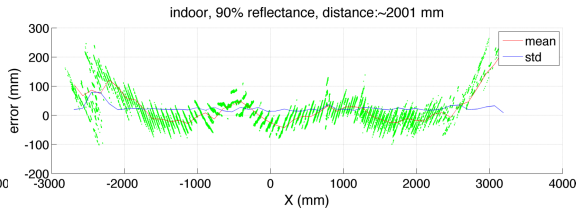
(C.2q)



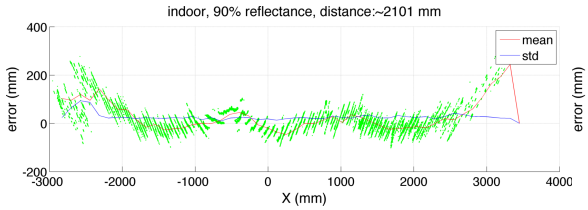
(C.2r)



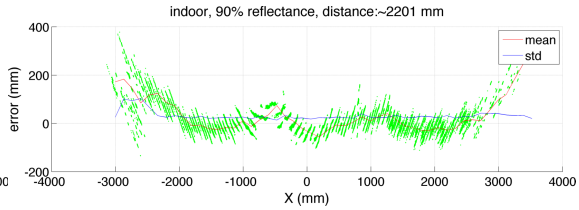
(C.2s)



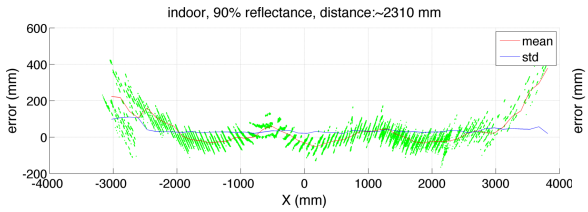
(C.2t)



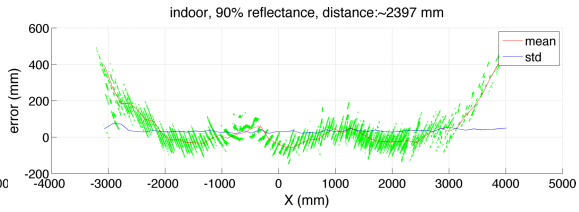
(C.2u)



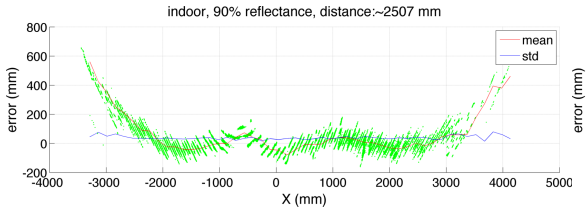
(C.2v)



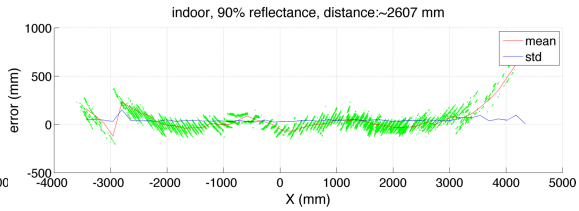
(C.2w)



(C.2x)



(C.2y)



(C.2z)

C.3 Angular resolution

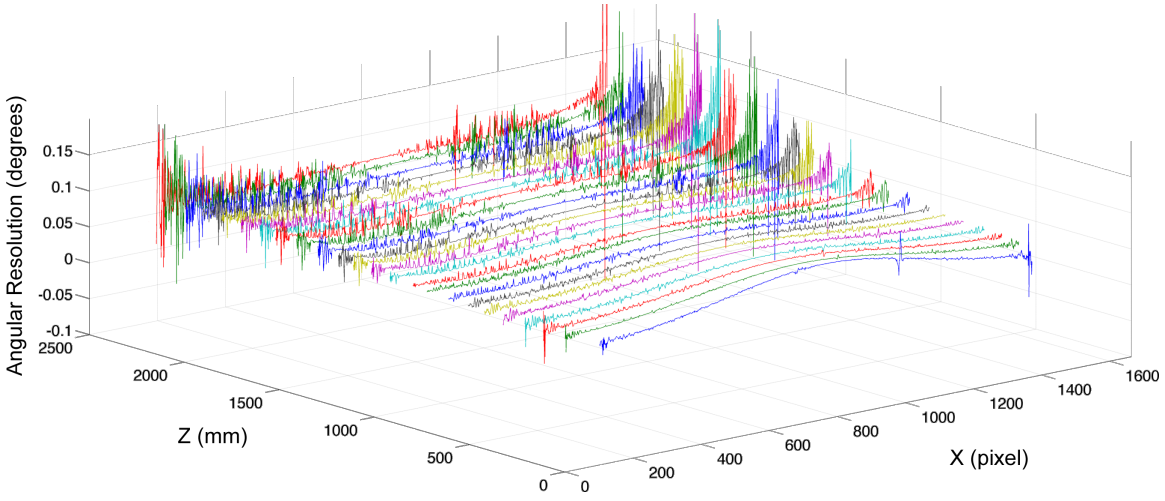


Figure C.3: Angular resolution for all measurements in the indoor-dataset. Resolution is averaged for all measurements in a set, aligned to the horizontal pixel position which produced the projection.

C.4 VideoCore IV load

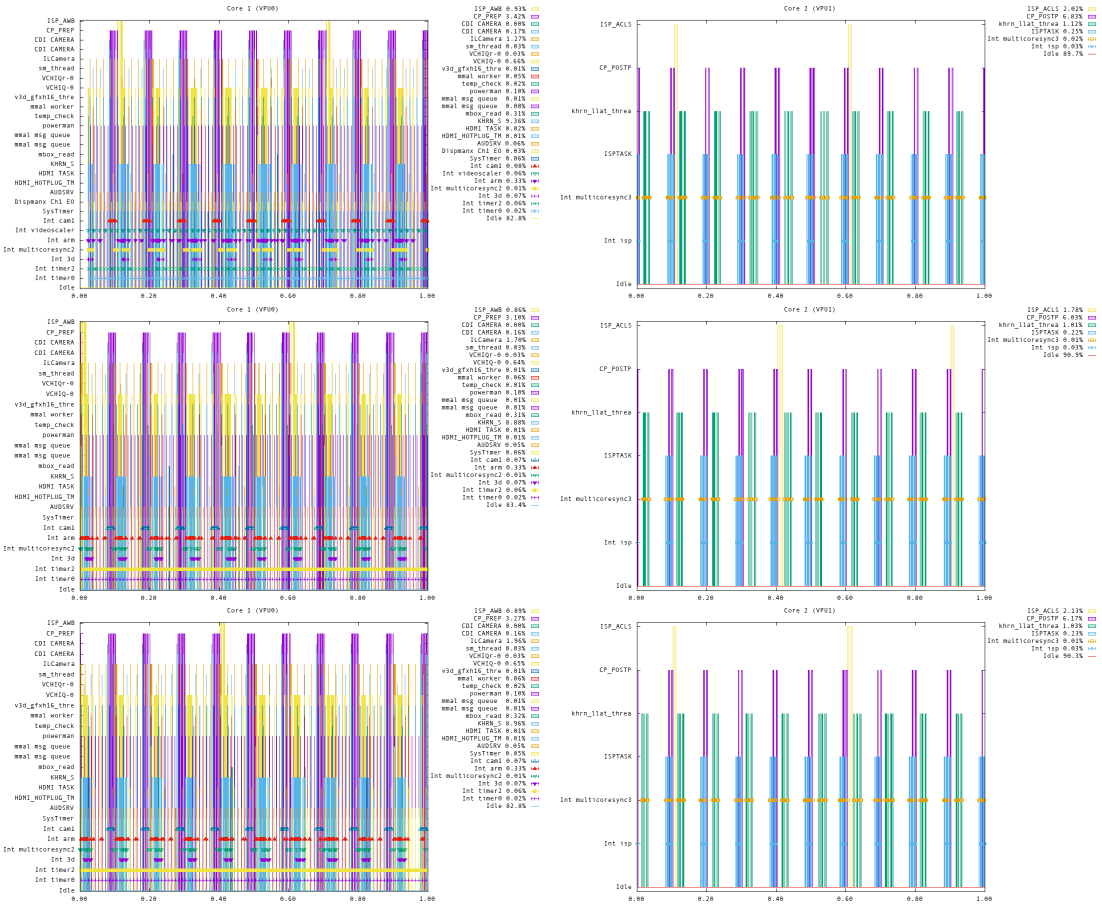


Figure C.4: Several one-second traces of the VPU cores. From the traces, the regular intervals of the fetching and processing of the captured frames can be observerd. The framerate is set to 10Hz. While being idle most of the time, the largest amount of processing power is dedicated to the image filtering (khrn*, v3d*, dispmanx*) and image signal pipeline (isp*). The mmal*, vchi* and mbox* threads manage the CPU-VPU communication.

Appendix D

Derivations and Proofs

Several proofs and derivations have been made in this thesis. This chapter shows the assumptions, schematics, and steps taken to get to the resulting set of formulas.

D.1 Triangulation

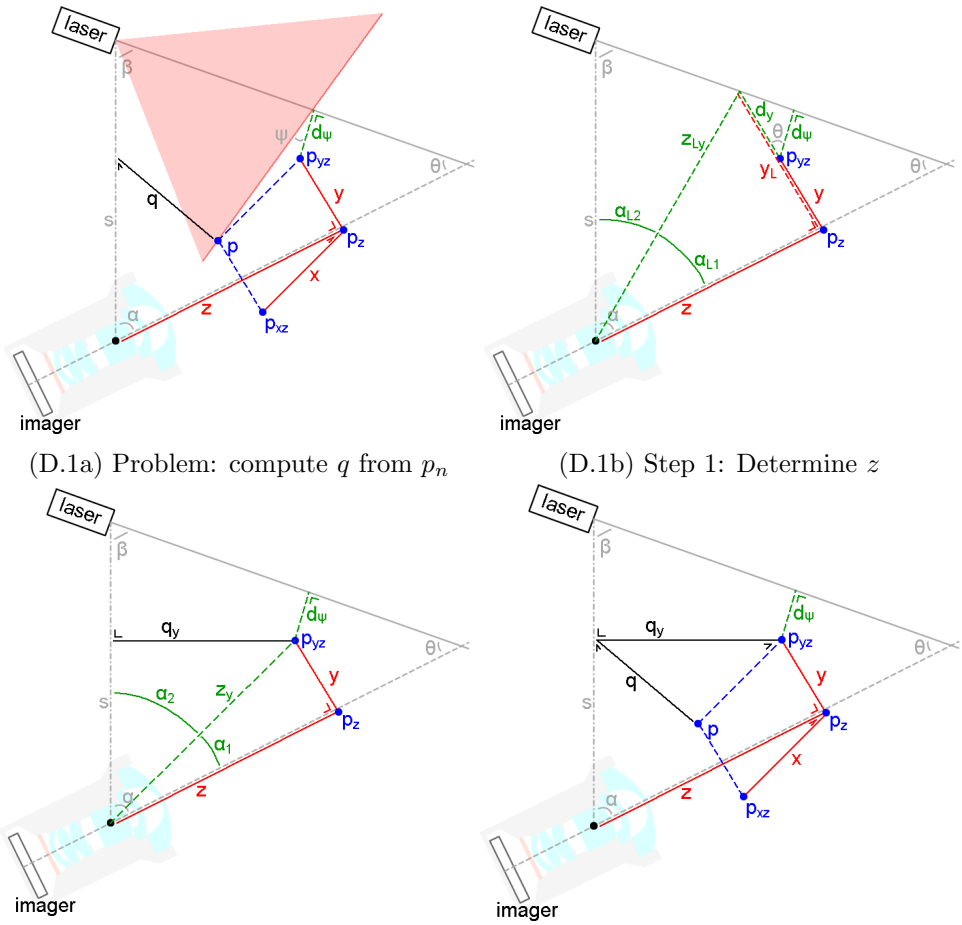
An important part of this thesis is the introduction of a generalized triangulation model, which is able to use both line- and spot- lasers to estimate an object distance q. This section shows the derivations of the introduced triangulation equations. First, the full derivation of a line-laser setup is given. Following this setup, a simplification of a generalized spot-laser system is given, with a concluding proof showing how the generalized model compares with the model used by Konolige et. al [29].

D.1.1 Line laser

The line-laser problem is shown in Figure D.1a: a line laser is placed at an angle β in a triangle with the optical center of a camera located at a distance s, tilted with an angle α. The optical axis of the camera crosses the central axis of the laser at an angle θ. The laser is tilted around its central axis at an angle ψ. When $\psi = .5\pi$, the laser-line is projected horizontally. Within the camera reference frame, an object reflects the laser at point $p = [x\ y\ z]^T$, where the distance towards this point is represented by q. In the camera, p is projected onto pixel position p_p. Using a camera model as shown in Chapter A, the position p_p can be transformed into $p_n = [x_n\ y_n\ 1]^T = [\frac{x}{z}\ \frac{y}{z}\ \frac{z}{z}]^T$, which represents the normalized projection of p. Hence the problem can be stated as:

Using the triangulation parameters $[\alpha, \beta, \theta, \psi, s]$, the distance q, from s towards p, needs to be computed, using the normalised value p_n.

Computing q is done in several steps. First, we derive an estimated value of z, followed by determining q_y and finally q.



(D.1a) Problem: compute q from p_n

(D.1b) Step 1: Determine z

(D.1c) Step 2: Determine q_y

(D.1d) Step 3: Determine q

Figure D.1: Steps to determine q. All lines and points in Figure D.1b and D.1c lay within the ZY-plane. The laser, q and the x-axis in Figure D.1a and D.1d are in depth.

Determining z: Using Figure D.1b, the z position of a point p in the camera reference frame can be computed. First, the length of y_L needs to be determined, which is the projection of y onto the central axis of the laser.

$$
\begin{aligned}
y_L &= y + d_y && && \text{(D.1)}\\
&= y + \frac{d_\psi}{cos(\theta)} && && \text{(D.2)}\\
&= y + \frac{\frac{x}{tan(\psi)}}{cos(\theta)} && \text{effect of } \psi \text{ on the x-axis} && \text{(D.3)}\\
&= y + \frac{x}{tan(\psi) * cos(\theta)} && && \text{(D.4)}\\
&= (y_n * z) + \frac{x_n * z}{tan(\psi) * cos(\theta)} && && \text{(D.5)}\\
&= z * \left(y_n + \frac{x_n}{tan(\psi) * cos(\theta)} \right) && && \text{(D.6)}
\end{aligned}
$$

With y_L, the angle towards the projection on the central axis of the laser can be computed, which in turn allows determining z_{Ly}:

$$
\begin{aligned}
\alpha_{L1} &= tan^{-1}\left(\frac{y_L}{z}\right) && && \text{(D.7)}\\
&= tan^{-1}\left(\frac{z * \left(y_n + \frac{x_n}{tan(\psi)*cos(\theta)}\right)}{z}\right) && \text{inserting (D.6)} && \text{(D.8)}\\
&= tan^{-1}\left(y_n + \frac{x_n}{tan(\psi) * cos(\theta)}\right) && && \text{(D.9)}\\
& && && \text{(D.10)}
\end{aligned}
$$

$$
\begin{aligned}
z_{Ly} &= sin(\beta) * \frac{s}{sin(\pi - \beta - \alpha_{L2})} && && \text{(D.11)}\\
&= sin(\beta) * \frac{s}{sin(\pi - \beta - (\alpha - \alpha_{L1}))} && && \text{(D.12)}\\
&= sin(\beta) * \frac{s}{sin(\pi - \beta - \alpha + \alpha_{L1})} && && \text{(D.13)}\\
&= s * \frac{sin(\beta)}{sin(\theta + \alpha_{L1})} && && \text{(D.14)}
\end{aligned}
$$

Finally, z_{Ly} is used to determine z:

$$z = cos(\alpha_{L1}) * z_{Ly} \tag{D.15}$$
$$= cos(\alpha_{L1}) * s * \frac{sin(\beta)}{sin(\theta + \alpha_{L1})} \quad \text{inserting (D.14)} \tag{D.16}$$
$$= s * \frac{sin(\beta)cos(\alpha_{L1})}{sin(\theta + \alpha_{L1})} \tag{D.17}$$
$$= s * \frac{sin(\beta)cos(tan^{-1}(...))}{sin(\theta + tan^{-1}(...))} \quad \text{inserting (D.6)} \tag{D.18}$$
$$= s * \frac{sin(\beta)\frac{1}{\sqrt{(...)^2+1}}}{\frac{sin(\theta)+(...)cos(\theta)}{\sqrt{(...)^2+1}}} \quad \text{resolving } tan^{-1} \tag{D.19}$$
$$= s * \frac{sin(\beta)}{sin(\theta) + (...)cos(\theta)} \tag{D.20}$$
$$= s * \frac{sin(\beta)}{sin(\theta) + \left(y_n + \frac{x_n}{tan(\psi)*cos(\theta)}\right) cos(\theta)} \tag{D.21}$$
$$= s * \frac{sin(\beta)}{sin(\theta) + y_n * cos(\theta) + \frac{x_n}{tan(\psi)}} \tag{D.22}$$

Determining q_y: Knowing z, the projection of q on the ZY-plane, q_y, can be determined. For this derivation Figure D.1c is used.

$$z_y = \frac{z}{cos(\alpha_1)} \tag{D.23}$$
$$= \frac{z}{cos(tan^{-1}(\frac{y}{z}))} \tag{D.24}$$
$$= \frac{z}{cos(tan^{-1}(y_n))} \quad \text{normalisation of } y \tag{D.25}$$
$$= \frac{z}{\frac{1}{\sqrt{y_n^2+1}}} \quad \text{resolving } cos * tan^{-1} \tag{D.26}$$
$$= z * \sqrt{y_n^2 + 1} \tag{D.27}$$

$$q_y = sin(\alpha_2) * z_y \tag{D.28}$$
$$= sin(\alpha_2) * (z * \sqrt{y_n^2 + 1}) \quad \text{inserting (D.27)} \tag{D.29}$$
$$= sin(\alpha - \alpha_1) * (z * \sqrt{y_n^2 + 1}) \tag{D.30}$$
$$= sin(\alpha - tan^{-1}\left(\frac{y}{z}\right)) * (z * \sqrt{y_n^2 + 1}) \tag{D.31}$$

$$= sin(\alpha - tan^{-1}(y_n)) * (z * \sqrt{y_n^2 + 1}) \qquad \text{normalisation of } y \tag{D.32}$$

$$= \left(\frac{sin(\alpha) - y_n * cos(\alpha)}{\sqrt{y_n^2 + 1}} \right) * (z * \sqrt{y_n^2 + 1}) \qquad \text{resolving } sin(tan^{-1}) \tag{D.33}$$

$$= z * (sin(\alpha) - y_n * cos(\alpha)) \tag{D.34}$$

Determining q As q_y is a projection of q and as the the x-axis is perpendicular to this projection, the distance towards p, q can be computed by:

$$q = \sqrt{q_y^2 + x^2} \tag{D.35}$$

$$= \sqrt{(z * (sin(\alpha) - y_n * cos(\alpha)))^2 + (x_n * z)^2} \tag{D.36}$$

$$= \sqrt{z^2 * (sin(\alpha) - y_n * cos(\alpha))^2 + z^2 * x_n^2} \tag{D.37}$$

$$= z * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2 + x_n^2} \tag{D.38}$$

Eventually, (D.22) could be inserted, resulting in the computation of q as:

$$q = s * \frac{sin(\beta) * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2 + x_n^2}}{sin(\theta) + y_n * cos(\theta) + \frac{x_n}{tan(\psi)}} \tag{D.39}$$

D.1.2 Spot laser

When a spot-laser is used, ψ does not has any effect: the projected spot stays a spot, irregardless how much the laser is rotated around its axis. Additionally, when we assume that the laser is aligned in the ZY plane of the setup, x_n should have no effect at all on the distance estimation. Therefore (D.39) can be rewritten as:

$$q = s * \frac{sin(\beta) * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2 + x_n^2}}{sin(\theta) + y_n * cos(\theta) + \frac{x_n}{tan(\psi)}} \qquad \text{duplicate of (D.39)} \tag{D.40}$$

$$= s * \frac{sin(\beta) * \sqrt{(sin(\alpha) - y_n * cos(\alpha))^2}}{sin(\theta) + y_n * cos(\theta)} \tag{D.41}$$

$$= s * \frac{sin(\beta) * (sin(\alpha) - y_n * cos(\alpha))}{sin(\theta) + y_n * cos(\theta)} \tag{D.42}$$

$$= s * \frac{sin(\beta)sin(\alpha) - y_n * sin(\beta)cos(\alpha))}{sin(\theta) + y_n * cos(\theta)} \tag{D.43}$$

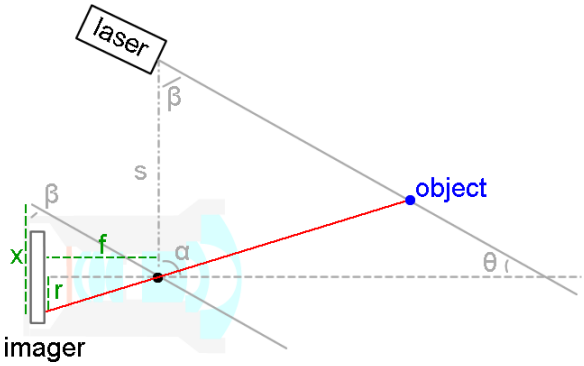


Figure D.2: Setup with $\alpha = .5\pi$ from [29].

D.1.3 Verification with the model of Konolige et al [29]

Konolige et al. describe in their paper the simplified model:

$$q = \frac{s * f}{x_p} \tag{D.44}$$

Where q and s are the earlier defined parameters: distance-to-object and distance-between-laser-camera. The other parameters, f, and x_p are the focal point of the lens (distance imager-optical center) and the difference between the pixel position of the reflected laser and the ray through the optical center, parallel to the laser respectively. An overview is shown in Figure D.2.

Their model does not correct for laser misalignment, hence (D.43) can be used to show our generalized model can be simplified to (D.44).

The first simplification which can be made is the assumption of $\alpha = .5\pi$, allowing to rewrite (D.43) as:

$$q = s * \frac{sin(\beta)sin(\alpha) - y_n * sin(\beta)cos(\alpha))}{sin(\theta) + y_n * cos(\theta)} \qquad \text{duplicate of (D.43)} \tag{D.45}$$

$$= s * \frac{sin(\beta)sin(.5\pi) - y_n * sin(\beta)cos(.5\pi)}{sin(\theta) + y_n * cos(\theta)} \tag{D.46}$$

$$= s * \frac{sin(\beta)}{sin(\theta) + y_n * cos(\theta)} \tag{D.47}$$

As y_n is normalised over z and both z and f are in line with the optical axis, y_n can additionally be defined as:

$$y_n = \frac{r}{f} \tag{D.48}$$

Inserting in (D.47) yields:

$$q = s * \frac{sin(\beta)}{sin(\theta) + y_n * cos(\theta)} \quad \text{duplicate of (D.47)} \tag{D.49}$$

$$= s * \frac{sin(\beta)}{sin(\theta) + \frac{r}{f} * cos(\theta)} \quad \text{inserting (D.48)} \tag{D.50}$$

$$= s * \frac{f * sin(\beta)}{f * sin(\theta) + r * cos(\theta)} \tag{D.51}$$

$$= s * \frac{f * sin(\beta)}{\frac{cos(\theta)}{cos(\theta)} f * sin(\theta) + r * cos(\theta)} \tag{D.52}$$

$$= s * \frac{f * sin(\beta)}{cos(\theta) * f * \frac{sin(\theta)}{cos(\theta)} + r * cos(\theta)} \tag{D.53}$$

$$= s * \frac{f * sin(\beta)}{cos(\theta) * f * tan(\theta) + r * cos(\theta)} \tag{D.54}$$

$$= s * \frac{f * sin(\beta)}{cos(\theta) * (x_p - r) + r * cos(\theta)} \tag{D.55}$$

$$= s * \frac{f * sin(\beta)}{x_p * cos(\theta)} \quad \text{cancellation of } r \tag{D.56}$$

$$= s * \frac{f * sin(\beta)}{x_p * cos(\pi - \alpha - \beta)} \tag{D.57}$$

$$= s * \frac{f * sin(\beta)}{x_p * cos(.5\pi - \beta)} \tag{D.58}$$

$$= s * \frac{f * sin(\beta)}{x_p * sin(\beta)} \tag{D.59}$$

$$= \frac{s * f}{x_p} \tag{D.60}$$

D.2 Laser-plane projection

In Chapter 3.2 we show that it is possible to transform the normalized 2D backward projected pixel coordinates to a 3D position, by representing the laser as a plane in the camera reference frame. This section shows the derivations for these equations. Additionally, it shows how the triangulation parameters $(\alpha, \beta, \theta, \psi, s)$ can be derived from the laser-plane.

D.2.1 $2D \Rightarrow 3D$

As shown in Chapter 3.2, a plane in 3D space can be described as:

$$z = a * x + b * y + c \tag{D.61}$$

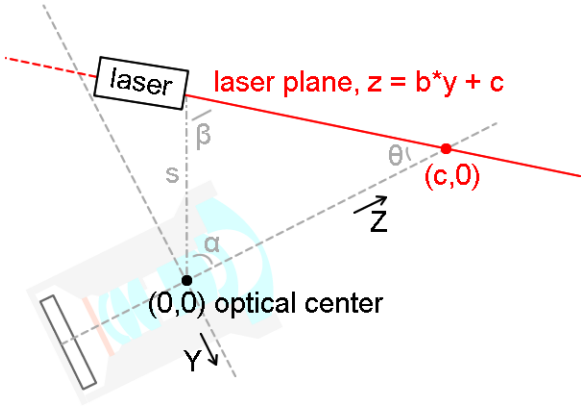


Figure D.3: Computing s and α from the laser plane, assuming β is known and $a = 0$.

After calibration, we know the laser-plane parameters (a_c, b_c, c_c) and from the backwards projection, we also know the normalised 2D position of the pixels, (x_n, y_n). This allows us to express the computation of z as:

$$z = a_c * x + b_c * y + c_c \tag{D.62}$$
$$= a_c * (x_n * z) + b_c * (y_n * z) + c_c \tag{D.63}$$
$$= z * (a_c * x_n + b_c * y_n) + c_c \tag{D.64}$$
$$1 = a_c * x_n + b_c * y_n + \frac{c_c}{z} \tag{D.65}$$
$$\frac{c_c}{z} = 1 - (a_c * x_n + b_c * y_n) \tag{D.66}$$
$$z = \frac{c_c}{1 - (a_c * x_n + b_c * y_n)} \tag{D.67}$$
$$\tag{D.68}$$

D.2.2 Laser-plane $\Rightarrow s, \alpha, \psi$

When we assume that β is known, the laser-plane parameters (a_c, b_c, c_c) can be used to make estimations of the triangulations setup. It should, however, be noted that such a computation is only valid under the assumption that all possible errors are modeled in the triangulation computation.

In our triangulation setup, s is defined as the distance from the optical centre of the camera - $(0, 0, 0)$ in the reference frame - to point $(0, y, z)$ at which the laser plane is crossed at an angle β. In Figure D.3 the YZ plane (with $x = 0$) is displayed. Using this schematic overview, we can compute θ and s as follow:

$$\theta = tan^{-1}\left(\frac{1}{b_c}\right) \tag{D.69}$$

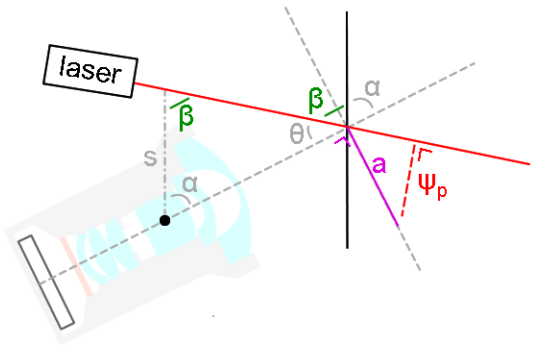


Figure D.4: Computing ψ from a_c. The black line is parallel to s. The grey-dotted lines are perpendicular to each other, where a_c is perpendicular to the optical axis of the camera and ψ_p perpendicular to the optical axis of the laser.

$$\frac{s}{sin(\theta)} = \frac{c_c}{sin(\beta)} \tag{D.70}$$

$$s = \frac{sin(\theta) * c_c}{sin(\beta)} \tag{D.71}$$

Using the same YZ plane, α can also be estimated:

$$\alpha = \pi - (\beta + \theta) \tag{D.72}$$

Computing the roll of the laser is a bit more challenging. Recalling Chapter D.1, ψ is defined orthogonal to the optical axis of the laser, whereas a_c from the fitted plane is defined orthogonal to the x-axis in the reference plane. An overview is shown in Figure D.4.

When assuming $y = 0$, the rotation of the plane around the x-axis is described with $z = a_c * x$. Note that it is not depending on c as this only provides an offset.

Computing ψ is done by assuming an unit step in the x-direction: from the camera reference frame the plane has changed by a factor a_c and the laser rotation results in an offset ψ_p as shown in Figure D.4. Using this step, ψ_p can be computed with:

$$\psi_p = sin(\angle_a) * a_c \tag{D.73}$$

$$= sin(\beta - (0.5\pi - \alpha)) * a_c \tag{D.74}$$

$$= -a_c * cos(\theta) \tag{D.75}$$

As ψ_p is the unit-step result for ψ, we can compute:

$$\psi = 0.5 * \pi - tan^{-1}(\psi_p) \tag{D.76}$$

$$= 0.5 * \pi - tan^{-1}(-a_c * cos(\theta)) \tag{D.77}$$