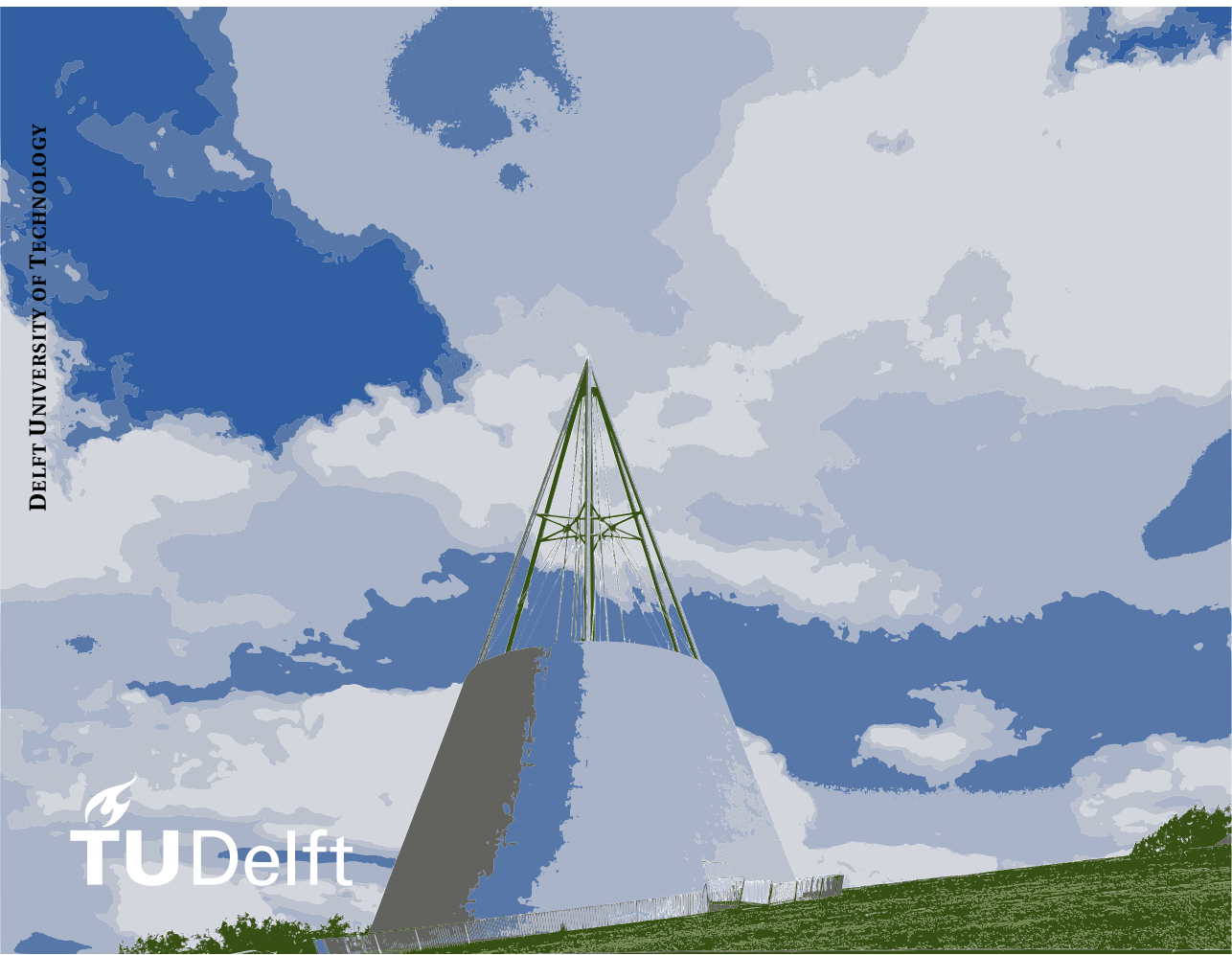
1D-TOF:

System modeling
Processing algorithm design
Implementation



Delft University of Technology

TU Delft

1D-TOF: SYSTEM MODELING, PROCESSING ALGORITHM DESIGN AND IMPLEMENTATION

Master of Science Dissertation

To obtain the degree of Master of Science,
track: Microelectronics,
at the Delft University of Technology,
to be defended publicly on August 19, 2022.

by

Mingzhe CHEN

Student number: 5222311,
Faculty Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, Delft, the Netherlands,
born in Hebei, China.

This dissertation has been approved by the

Thesis advisor (chair): Prof. Dr. Ir. Albert Theuwissen
Thesis co-supervisor: Dr. Esteban Venialgo Araujo
Company supervisor: Kezheng Ma

Composition of thesis committee:

Prof. Dr. Ir. Albert Theuwissen,	Technische Universiteit Delft
Dr. Padmakumar Rao,	Technische Universiteit Delft
Dr. Esteban Venialgo Araujo,	Technische Universiteit Delft
Prof. Dr. Ir. Guy Meynants,	Katholieke Universiteit Leuven
Other members:	
Kezheng Ma,	Silicon Integrated B.V.
Ting Gong,	Silicon Integrated B.V.

Keywords: LIDAR, 1D-TOF, D-TOF, depth sensing, target ranging

Printed by: Proefschriftenprinten

Front & Back: TU Delft library close-up view that is photographed and post-processed by me.

ISBN 000-00-0000-000-0

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

SUMMARY

Light detection and ranging (LIDAR) is rapidly emerging as a key technology for depth sensing applications. LIDAR acts as an auxiliary ranging technique, such as radar, thermal image ranging, image recognition, and time-of-flight (TOF) sensor. As a special type of direct time-of-flight (D-TOF) system, this work presents the system design and the hardware implementation of a one dimension time-of-flight (1D-TOF) sensor that uses LIDAR with single-photon avalanche diodes (SPADs). The system is designed for consumer electronics, serving for short-distance target ranging in proximity and face detection.

A typical 1D-TOF LIDAR system is composed of a pulsed laser which emits photons, and a sensor which measures the photons that are reflected back from the target. In 1D-TOF LIDAR systems, the following elements are of great importance: SPAD, time-to-digital converter (TDC), TOF histogram, and depth estimation algorithm.

SPAD is the key component of the system, which is designed to detect a single photon. The target ranging can be estimated by measuring the travel time of the photons. With the help of SPADs, the detection of the photons is converted into a response of the SPAD circuity, in which the optical signal is converted to the electrical signal. And, we can measure the moment in which the electrical signal is active to estimate the travel time of the photons.

TDC is a tool for time measurement. In a 1D-TOF LIDAR system, it measures the exact moment, when a SPAD is triggered by a single photon. The measured time is called the timestamp of the photon.

A TOF histogram accumulates the timestamps of the photons at every time interval. In a TOF histogram, we can observe the time distribution of the photons. In essence, the photons come from two sources, noise and signal. With the help of algorithms, we can derive signal information and reject noise information from a TOF histogram. Also, the algorithm enables us to retrieve the TOF information of the histogram, in which the distance between the target and the sensor lies within.

In this work, the basic concepts of the 1D-TOF LIDAR system is described in chapter 1, and a prototype architecture is discussed in chapter 2. Next, the design is decomposed into stages, including system modeling in chapter 3, system trade-off analysis in chapter 4, algorithm design in chapter 5 and hardware implementation in chapter 6. Moreover, we also propose novel approaches both in system level modeling and algorithm design based on artificial intelligence methods.

CONTENTS

NOMENCLATURE

1D-TOF	One dimension time-of-flight
AEL	Accessible emission limits
ANN	Artificial neural network
CvUT	Count-vector under test
D-TOF	Direct time-of-flight
DCR	Dark count rate
DE	Depth estimator
DSP	Digital signal processor
FC	Failure condition
FF	Fill factor
FOI	Field-of-illumination
FOV	Field-of-view
FPGA	Field-programmable gate array
FRA	Failure rate analysis
GUI	Graphic user interface
HB	Histogram builder
ILA	Integrated-logic-analyzer
LIDAR	Light detection and ranging
LSB	Least-significant-bit
LUT	Look-up-table
LVDS	Low-voltage differential signal
MŜE	Sample mean-square-error
MSE	Mean-square-error
PCB	Printed circuit board

PD	Peak detection
PDF	Probability density function
PDP	Photon detection probability
PLL	Phase lock loop
RAM	Random-access memory
ROM	Read-only memory
RTL	Register-transfer level
RV	Random variable
SPAD	Single-photon avalanche diodes
TCSPC	Time-correlated single-photon counting
TDC	Time-to-digital converter
TOF	Time-of-flight
VCSEL	Vertical-cavity surface-emitting Laser

1

INTRODUCTION

Mingzhe CHEN

This chapter focuses on the introduction of a one dimension time-of-flight (1D-TOF) light detection and ranging (LIDAR) sensor system. Several utilized devices and techniques are presented and introduced, such as SPAD, time-to-digital converter (TDC) and timestamp processing mechanism. The aim is to provide the reader with a general understanding of the basic technical background of this work.

1.1. LIDAR WITH 1D-TOF SENSOR

In general, direct time-of-flight (D-TOF) sensors can be classified into two types according to different application fields. For sensors that focus on depth imaging, they are categorized as TOF imaging sensors; for sensors that only serve for target ranging with single depth value, they are categorized as 1D-TOF sensors. In this work, we focus on the study of a 1D-TOF LIDAR sensor system.

The proposed 1D-TOF sensor targets consumer electronics applications. This system has a laser source as a transmitter and a SPAD sensor as a receiver. The laser source emits a light pulse, which arrives at a target. The light is reflected back by the target, captured and received by the SPAD sensor. By measuring the time difference between the transmitted pulse and received pulse, the distance is calculated based on how long the light has traveled. Normally, the system will do multiple measurements based on time-correlated single-photon counting (TCSPC) cycles, in order to form a histogram with timestamps. By locating the peak of the histogram, the TOF of photons traveling from the transmitter to the receiver is known. However, the shape of the histogram is heavily affected by noise sources like dark count, ambient light and SPAD cross-talk. As a result, finding the peak of the histogram is not a straight forward task. System modeling and

algorithm research is needed in order to design a system that fulfills the performance requirements.

1.2. SINGLE-PHOTON AVALANCHE DIODE

Single-photon avalanche diodes (SPADs) are p-n junctions that are designed to perform their functions biased above their breakdown voltage, where a high electric field is utilized to detect a single photon with high sensitivity and low timing jitter [1].

In SPAD operation mode, a detected photon generates a photocarrier in the depletion region, which can initiate an avalanche breakdown through impact ionization, generating avalanche current [2].

Quenching and recharge mechanisms are followed after the SPAD enters into breakdown. The avalanche must be stopped before the device overheats. In general, there are two types of quenching and recharge techniques: passive and active. Once the avalanche is quenched, the SPAD needs to be biased above its breakdown voltage again, in which the passive or active mechanism takes place [3].

1.3. TIME-TO-DIGITAL CONVERTER

A very important block in TOF sensor systems is the time-to-digital converter (TDC). In this context, it is used for measuring the arrival time of photons reflected by the target object or noise sources. TDCs, which are included into a per-pixel architecture [4] or shared architecture [5], are also implemented in TOF LIDAR systems. Due to area and power trade-offs, shared-TDC architectures are popular choices for TOF LIDAR system design.

1.4. TIMESTAMP SIGNAL PROCESSING

A timestamp is an output code generated by the TDC, related to the arrival time of a single photon. Due to the TDC's least-significant-bit (LSB), the recorded timestamp can only be integer multiples of it. Typically, TOF systems require the accumulation of TCSPC cycles to build a histogram. Multiple timestamps are acquired, and the timestamps related to signal arrival can be modelled as a Gaussian, located among a certain area of the histogram, showing a shape of a peak. The timestamps related to noise arrival are exponentially distributed, showing a noise floor. This gives designers a intuitive understanding of how the sensor works. In addition, histogram processing algorithms can be applied, in order to minimize the influence from background noise, to find the relation between histogram peaks and real distance.

BIBLIOGRAPHY

[1] Matthew W Fishburn. *Fundamentals of CMOS single-photon avalanche diodes*. fishburn, 2012.

[2] Chockalingam Veerappan. "Single-Photon Avalanche Diodes for Cancer Diagnosis". PhD thesis. Delft University of Technology, 2016. ISBN: 978-94-6186-620-2. DOI: 10.4233/uuid:7db13e84-9ced-4c9c-94fa-8c2a14ad6679.

[3] Ryan M. Field, Simeon Realov, and Kenneth L. Shepard. "A 100 fps, Time-Correlated Single-Photon-Counting-Based Fluorescence-Lifetime Imager in 130 nm CMOS". In: *IEEE Journal of Solid-State Circuits* 49.4 (2014), pp. 867–880. DOI: 10.1109/JSSC.2013.2293777.

[4] Chockalingam Veerappan et al. "A 160×128 single-photon image sensor with on-pixel 55ps 10b time-to-digital converter". In: *2011 IEEE International Solid-State Circuits Conference*. 2011, pp. 312–314. DOI: 10.1109/ISSCC.2011.5746333.

[5] Augusto Ronchini Ximenes et al. "A 256×256 45/65nm 3D-stacked SPAD-based direct TOF image sensor for LiDAR applications with optical polar modulation for up to 18.6dB interference suppression". In: *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*. 2018, pp. 96–98. DOI: 10.1109/ISSCC.2018.8310201.

2

1D-TOF ARCHITECTURE

Mingzhe CHEN

For a 1D-TOF system, it is desirable to achieve a large maximum measurable distance or distance range. Accuracy, speed and power are common requirements, while the system must ensure its proper function [1]. To meet these requirements, many architectures for a TOF system are proposed. In this section, a general 1D-TOF system-level architecture is introduced, which is meant to be the prototyping basis for building more complex models and processing techniques.

2.1. PROPOSED ARCHITECTURE

A simplified block diagram of the proposed architecture is shown in Fig. 2.1. It is a prototype architecture, which is used for understanding the data flow, in order to build models, do simulations and apply algorithms. Also, this architecture is the previous prototyping step before designing a fully integrated 1D-TOF sensor. It is conceived with three major modules, including a simplified sensor, field-programmable gate array (FPGA) and acquisition computer. Each block will be presented in the following sections in detail.

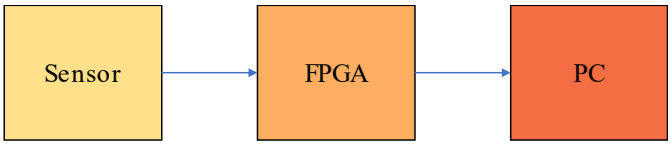


Fig. 2.1: Simplified block diagram of the proposed 1D-TOF system.

2.2. SENSOR MODULE

The sensor module mainly includes the SPAD and a basic digital circuitry, as shown in Fig. 2.2. The laser is integrated into the sensor module, which emits photons to the target. The photons are reflected by the target and will generate an avalanche current when they are detected by the SPADs. The SPADs are controlled by their quenching and recharge circuitry, to ensure their continuous functional use and to protect them from overheating. In addition, a buffer, a comparator, and a filter block are implemented into an FPGA. Each SPAD and its circuitry can be controlled independently, so we can choose the number of SPAD that become active to receive photons.

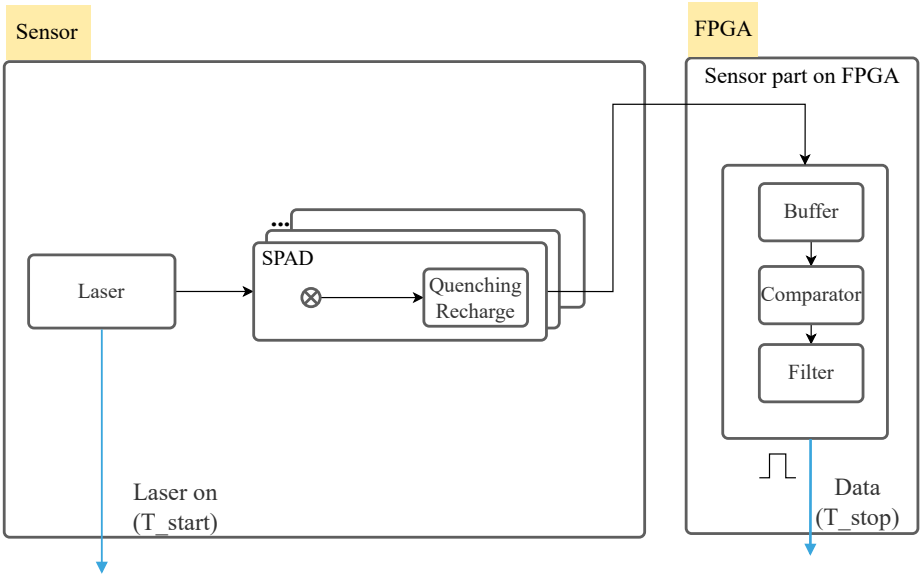


Fig. 2.2: Sensor module in the proposed 1D-TOF system.

2.3. FPGA MODULE

The FPGA module is shown in Fig. 2.3. It has an essential block, which is the TDC, that generates a timestamp after receiving a start signal from the laser and a stop signal from the sensor part on FPGA. In the proposed architecture, only one TDC is considered in order to study the trade-offs before increasing the system complexity. The stored time-stamps are used to build a histogram via the histogram builder, which is connected to a digital signal processor (DSP) (see Fig. 2.3). The function of the DSP block is calculating the target distance from the histogram. Random-access memory (RAM) is used to store either intermediate data for debugging or the final value for output readout (see Fig. 2.3), as well as the histogram data. In the prototype design stage, the RAM should store the complete histogram, and let other algorithm processing blocks to read the histogram

data from it[1].

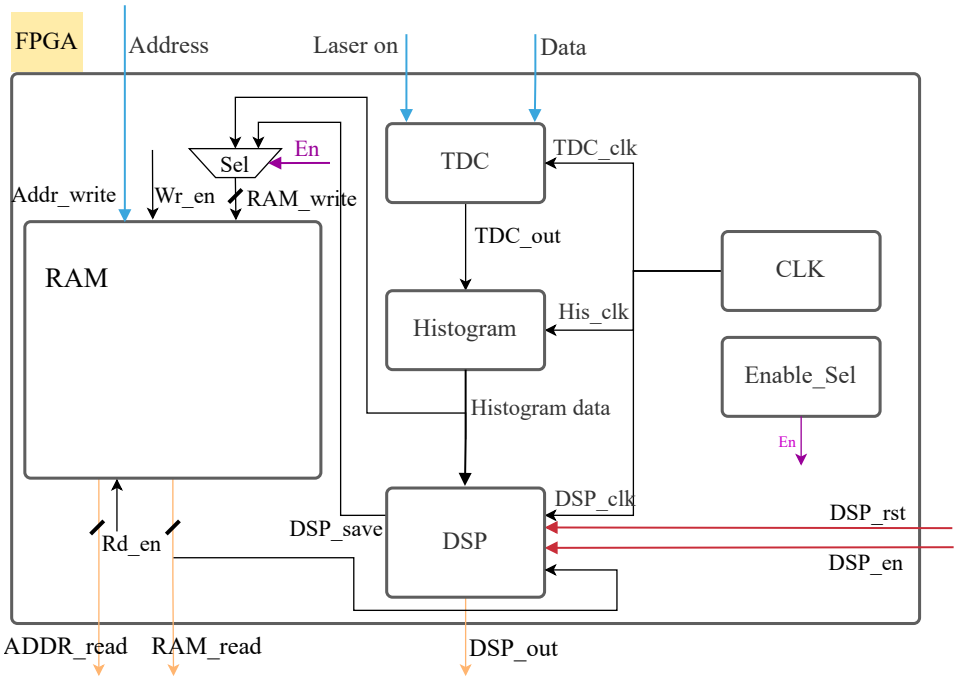


Fig. 2.3: FPGA module in the proposed 1D-TOF system.

2.4. ACQUISITION COMPUTER MODULE

The acquisition computer block, which is shown in Fig. 2.4, has three main functions: control, readout, and displaying through a graphic interface. All control signals for the modules and submodules are generated from a control block (see Fig. 2.4), including DSP_rst, DSP_en signals and Laser_control, Mode_selector, CLK_control signals. The readout block can read the FPGA RAM content and convert its data into readable information. Or, send it to the graphic user interface (GUI) for displaying purposes, which is convenient for debugging the system.

2.5. DISCUSSION

To summarize, the proposed architecture is a prototype for model analysis, simulation and algorithm implementation in later parts of the project. It is expected to use the results of this work for future integrated sensor analysis and design.

[1] In order to save area, there are also partial-histogram techniques being applied, in which way the RAM only needs to store a partitioned inter-frame histogram [2].

2

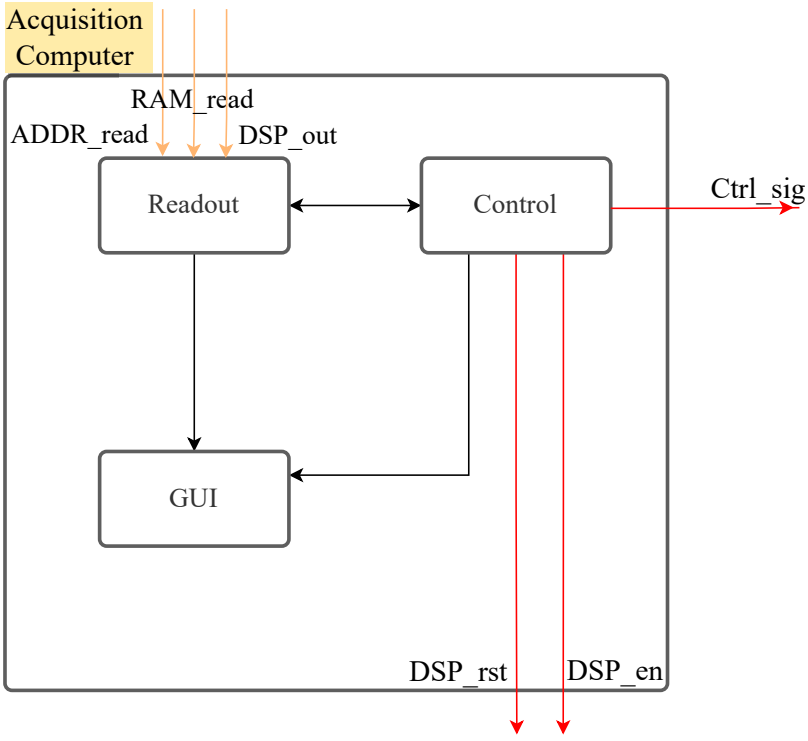


Fig. 2.4: Acquisition computer module in the proposed 1D-TOF system.

BIBLIOGRAPHY

[1] Augusto Ronchini Ximenes et al. “A 256×256 45/65nm 3D-stacked SPAD-based direct TOF image sensor for LiDAR applications with optical polar modulation for up to 18.6dB interference suppression”. In: *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*. 2018, pp. 96–98. DOI: 10.1109/ISSCC.2018.8310201.

[2] Ion Vornicu et al. “Compact Real-Time Inter-Frame Histogram Builder for 15-Bits High-Speed ToF-Imagers Based on Single-Photon Detection”. In: *IEEE Sensors Journal* 19.6 (2019), pp. 2181–2190. DOI: 10.1109/JSEN.2018.2885960.

3

LIDAR SYSTEM MODELING

Mingzhe CHEN

As mentioned in the introduction chapter, 1D-TOF LIDAR sensor is a special type of the D-TOF LIDAR sensor, which focuses on target ranging, only. To study the proposed 1D-TOF LIDAR sensor performance and evaluate several depth estimation methods, we start with a simulation and signal processing flow that models general D-TOF LIDAR systems. In this chapter, the optical modeling stage estimates the detected laser photons per TCSPC cycle. Also, the noise rate, in terms of counts per second, is calculated in this stage. The noise events include spurious counts due to background light as well as dark counts. Next, the time statistics modeling stage simulates the timing behavior of the signal and noise photons. In the following sections, details about the optical and time-statistics modelings are described and discussed.

3.1. EYE SAFETY IN LIDAR SYSTEM

In a D-TOF LIDAR system, a laser source is required to emit energy to the target. As this system is designed for consumer electronics applications, it is very important to keep the energy under eye safety levels. The goal is to have a Class 1 LIDAR system that is safe to operate under reasonably foreseeable conditions, including the intrabeam viewing [1] to the laser source [1].

Essentially, the wavelength of the light source is predefined as 940 nm, and the exposure time (T_{exp}) for one measurement is the product of the laser repetitive period ($1/f$) and the total TCSPC cycles. T_{exp} is given by

$$T_{\text{exp}} = \frac{1}{f}\text{TCSPC}. \tag{3.1}$$

[1] A viewing condition, in which the human eye is exposed to all or a part of the laser beam.

According to IEC 60825-1:2014, the maximum accessible emission limits (AEL) for typical Class 1 laser products can be summarized in table 3.1.

$\mathbf{T_{exp}}$ **[s]**	**Maximum AEL [J]**
$10^{-13} - 10^{-11}$	3.8×10^{-8}
$10^{-11} - 5 \times 10^{-6}$	7.7×10^{-8} C_4[1]
$5 \times 10^{-6} - 10$	7×10^{-4} $(T_{exp})^{0.75}$ C_4[1]

[1] C_4 is a correction factor. At 940 nm wavelength, C_4 is 3.02.

Table 3.1: Maximum AEL for Class 1 laser products [1].

Thus, the maximum average power ($P_{m,avg}$) of the laser source can be derived as

$$P_{m,avg} = \frac{AEL}{T_{exp}}. \tag{3.2}$$

Based on equation 3.2, we can set the laser power in the allowable range, regarding eye safety. In the following sections of this chapter, we first define the system specifications, and examine the laser source power level. And, we perform optical and time-statistics modelings with a safe laser source power.

3.2. OPTICAL MODEL

The optical simulation and signal processing flow can simulate a single D-TOF LIDAR scenario with fixed simulation parameters. It estimates the detected laser photons per TCSPC cycle (S_p) and the noise rate (N_r). And, as a result, it outputs a single value that corresponds to the estimated target depth ($\hat{D}$), which is calculated based on a timestamp histogram (**H**). However, in practice, we simulate several scenarios in the same run by sweeping the simulation parameters. So, we obtain a set of **H** and $\hat{D}$, in which their elements correspond to the different simulation parameters (see Fig. 3.1).

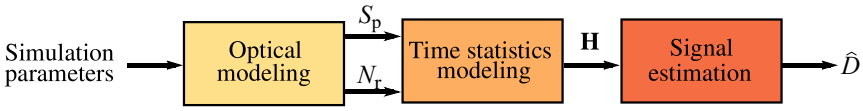


Fig. 3.1: Simulation flow of the proposed D-TOF LIDAR scenario.

Typically, in D-TOF LIDAR, the target surface is considered to be a perfect light diffuser (a Lambertian reflector). In order to simplify the simulation model, previous works assumed that the laser ray always perpendicularly hits the target surface [2, 3]. However, when the D-TOF LIDAR sensor moves closer to the target, the incidence angle of the light ray cannot be longer considered perpendicular (see $\theta_{e(i,j)}$ in Fig. 3.2(a)). And, this is the typical case in consumer electronics applications, particularly in proximity of face detection for smartphones.

Therefore, we propose a model, in which the target is divided into sub-elements, that are considered as individual Lambertian reflectors. Also, this pixelated model can consider situations in which the laser field-of-illumination (FOI) exceeds the target surface

area (see Fig. 3.2(a)). In this model, the detector consists of a lens, an optical bandpass filter and a SPAD array. The detector collects the summation of the diffused power from the pixelated sub-elements whose locations are inside the FOI of the laser, and the field-of-view (FOV) of the detector (see Fig. 3.2(b)).

The optical model is based on a pseudo-sequential ray tracing algorithm that first calculates light paths between the vertical-cavity surface-emitting laser (VCSEL) and the target surface. Next, it calculates the light paths between the sub-elements of the pixelated target surface and the SPAD sensor. Fig. 3.2 shows the representation of the D-TOF LIDAR scene in which the optical model is utilized.

In order to simulate a D-TOF system utilized for consumer electronics applications, the specifications of the system are presented in table 3.2, which are based on standard specifications of similar designs.

A laser power check is performed, in order to ensure eye safety. We assume that the TCSPC is 30,000. Thus, from the laser repetitive frequency, $P_{\mathrm{m,avg}}$ is approximated as

$$P_{\mathrm{m,avg}} = \frac{7\times10^{-4}(T_{\mathrm{exp}})^{0.75}\mathrm{C}_4}{\frac{1}{f}\times 30000} \approx 12.77\mathrm{mW}, \tag{3.3}$$

where the 7.36 mW average laser power (P_{s}) in table 3.2 is lower than $P_{\mathrm{m,avg}}$.

3.2.1. SIGNAL EVENT

A signal event is defined as a laser photon detection that generates an avalanche current in a SPAD. The signal light path is divided into two sub-paths:

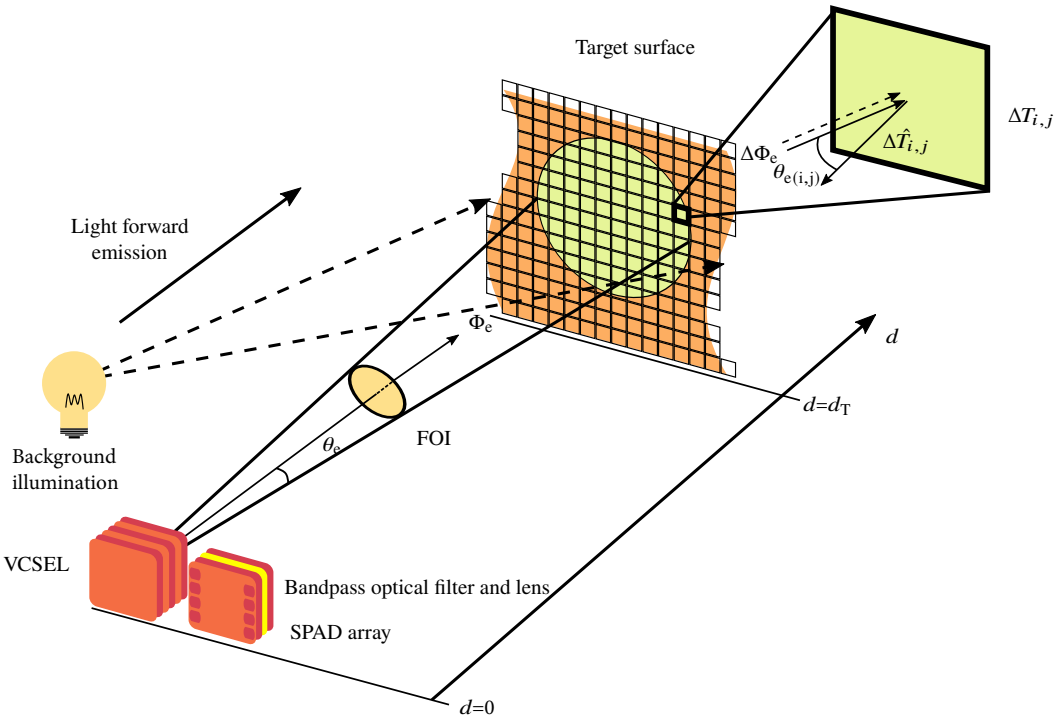
- the emission path in which laser source emits photons to the target (see Fig. 3.2(a));
- the reflection path in which the target reflects photons back onto the sensor lens (see Fig. 3.2(b)).

In the first sub-path, the amount of incident VCSEL light received by a sub-element of the pixelated target surface is calculated. The VCSEL and the sensor are placed at the beginning of the d-axis and are denoted as $d = 0$, in which the sensor contains a SPAD array, a bandpass filter and a lens. And we define d_{T} as the distance from the VCSEL surface to the target surface. The coverage sphere of the VCSEL is determined by a cone with its apex angle $2\theta_{\mathrm{e}}$, which is equal to the FOI (see Fig. 3.2(a)). The total solid angle that corresponds to the FOI (Ω) can be approximated as

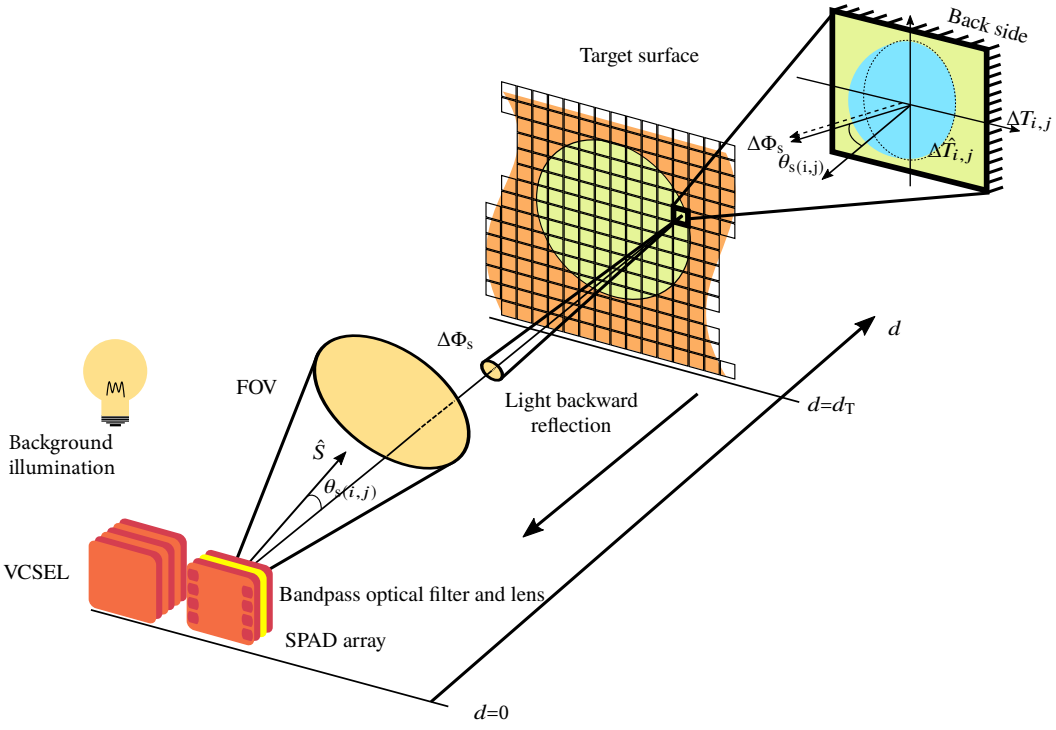
$$\Omega = \int_0^{2\pi}\int_0^{\theta_{\mathrm{e}}} \sin\theta\, d\theta\, d\phi \approx 4\pi\sin^2\frac{\theta_{\mathrm{e}}}{2} = 4\pi\sin^2\frac{\mathrm{FOI}}{4}. \tag{3.4}$$

At every sub-element of the pixelated target surface ($\Delta T_{i,j}$), we define $\theta_{\mathrm{e}(i,j)}$ as the incident angle between a VCSEL light ray and the normal vector to the surface of $\Delta T_{i,j}$ (see Fig. 3.2(a)). So, the distance $d_{\mathrm{e}(i,j)}$ from the VCSEL to $\Delta T_{i,j}$ is given by

$$d_{\mathrm{e}(i,j)} = \frac{d_{\mathrm{T}}}{\cos\theta_{\mathrm{e}(i,j)}}. \tag{3.5}$$



(a)



(b)

Fig. 3.2: Optical simulation setup. (a) Diagram of VCSEL light and noise emission. (b) Diagram of SPAD array's light detection.

	Parameter	Value	Note
Laser[1]	P_s	7.36 mW	Average power[4]
	λ	940 nm	Wavelength
	f	40 MHz	Repetitive frequency
	FOI	21°	Field-of-illumination
Noise[1]	E_v	14k lux	Halogen light illuminance
Target[1]	ρ	8 – 60%	Reflectivity
	L	26 cm	Length
	W	20 cm	Width
Lens[1]	d_l	11 mm	Lens diameter
	A_l	9.5×10^{-5} m^2	Lens area
	η_l	80%	Lens efficiency
Optical filter[2]	B_0	20 nm	Passband
	η_f	67%	Filter efficiency
Sensor[3]	M	40	Number of SPADs
	FOV	24°	Field-of-view
	PDP@940nm	1 – 2%	Photon detection probability
	FF	25%	Fill factor
	T_D	20 ns	SPAD deadtime

[1] The specifications are from custom's requirements given to Silicon Integrated (SI) B.V.
[2] The specifications are from the device available on Thorlabs [4].
[3] The specifications are defined by SI B.V.
[4] Unless otherwise specified, the power mentioned in this article is the average power.

Table 3.2: System specifications.

And the solid angle $\Delta\Omega_{(i,j)}$ from $\Delta T_{i,j}$ is approximated as

$$\Delta\Omega_{(i,j)} = \frac{\Delta\hat{T}_{i,j}}{d^2_{e(i,j)}}, \tag{3.6}$$

where $\Delta\hat{T}_{i,j}$ is the area of $\Delta T_{i,j}$.

When $\Delta T_{i,j}$ is located inside the FOI, a radiant flux of $\Delta\Phi_e$ is calculated as the ratio between the total solid angle Ω and the solid angle $\Delta\Omega_{(i,j)}$ and multiplied by the total VCSEL light power (P_s). This calculation can be expressed as

$$\Delta\Phi_{e(i,j)} = \begin{cases} \dfrac{P_s \Delta\hat{T}_{i,j}}{4\pi \sin^2(\frac{FOI}{4}) d^2_{e(i,j)}} & \theta_{e(i,j)} \leq \dfrac{FOI}{2} \\ 0 & \theta_{e(i,j)} > \dfrac{FOI}{2} \end{cases}. \tag{3.7}$$

The second sub-light-path of the ray tracing algorithm corresponds to the detected light by the D-TOF LIDAR sensor, which is back-reflected from the target surface. The

radiant flux $\Delta\Phi_{s(i,j)}$ emitted by the $\Delta T_{i,j}$ is the product of the received radiant flux and the reflectivity ρ of the target surface, and is defined by

$$\Delta\Phi_{s(i,j)} = \rho\Delta\Phi_{e(i,j)}. \tag{3.8}$$

The detected radiant intensity (ΔI_e) by the D-TOF LIDAR sensor, emitted from $\Delta T_{i,j}$, is proportional to the product of the peak radiant intensity I_0 and cosine $\theta_{s(i,j)}$, which is the angle between the normal vector of the sensor $\hat{S}$ and the normal vector of $\Delta T_{i,j}$ (see Fig. 3.2(b)). ΔI_e is given by

$$\Delta I_e = I_0 \cos\theta_{s(i,j)}. \tag{3.9}$$

A plane radiator or reflector that is perfectly diffusive emits light in all directions. And, the total emitted power is contained within half sphere with respect to the normal of that plane (see Fig. 3.2(b)). Therefore, the radiant flux $\Delta\Phi_{s(i,j)}$ diffused from $\Delta T_{i,j}$ of the target is calculated as the surface integral of the diffused radiant intensity ΔI_e over the solid angle $d\Omega$ of the half sphere:

$$\Delta\Phi_{s(i,j)} = \int_{\Omega} \Delta I_e d\Omega = \pi I_0. \tag{3.10}$$

According to the inverse-square law, the irradiance (ΔE_e), which is emitted from $\Delta T_{i,j}$, can be presented as

$$\Delta E_e = \frac{I_e}{r^2} = \frac{\Delta\Phi_{s(i,j)}\cos\theta_{s(i,j)}}{\pi d^2_{s(i,j)}}, \tag{3.11}$$

where $d_{s(i,j)}$ is the distance from the sub-element to the D-TOF LIDAR sensor [5].

The total radiant flux Φ_s sampled at the lens, whose area is A_l, is the summation of radiant flux $\Delta\Phi_{s(i,j)}$ of every $\Delta T_{i,j}$, but only if they are inside the FOV. Φ_s is calculated as follows:

$$\Phi_s = \sum_{i,j} \begin{cases} \dfrac{A_l\Delta\Phi_{s(i,j)}(\cos\theta_{s(i,j)})^2}{\pi d^2_{s(i,j)}} & \theta_{s(i,j)} \le \dfrac{\text{FOV}}{2} \\ 0 & \theta_{s(i,j)} > \dfrac{\text{FOV}}{2} \end{cases}. \tag{3.12}$$

By combining equations 3.7, 3.8 and 3.12, Φ_s can be directly calculated based on the parameters of the system:

$$\Phi_s = \sum_{i,j} \begin{cases} \dfrac{P_s\Delta\hat{T}_{i,j}\rho A_l(\cos\theta_{s(i,j)})^2}{4\pi^2\sin^2(\frac{\text{FOI}}{4})d^2_{e(i,j)}d^2_{s(i,j)}} & \theta_{e(i,j)} \le \dfrac{\text{FOI}}{2}, \theta_{s(i,j)} \le \dfrac{\text{FOV}}{2} \\ 0 & \text{otherwise} \end{cases}. \tag{3.13}$$

Next, we calculate S_p, which represents the photons per laser pulse. S_p considers the lens efficiency η_l. Also, it accounts for a reduction ratio $2/\pi$ since the light-sensitive area of the sensor is represented as a square, and it is inscribed in a circle that corresponds to the light projected by the lens [2]. An optical bandpass filter is placed between the lens and the D-TOF LIDAR sensor, which is used to reduce background ambient light (see

Fig. 3.2(b)). So, the filter efficiency $\eta_f(\lambda)$ is also considered as a power loss. In addition, it is assumed that there are in total M SPAD pixels in the uniform SPAD array. Therefore, the number of photons per TCSPC cycle can be calculated as

$$S_p = \Phi_s \eta_l \eta_f(\lambda) \frac{2\lambda}{M\pi f hc} \text{PDE}, \tag{3.14}$$

where h is the Plank constant, c is the light speed, f is the VCSEL repetitive frequency and λ is the VCSEL wavelength. The photon detection efficiency (PDE) is defined as the product of photon detection probability (PDP) and fill factor (FF) of the SPAD array.

3.2.2. NOISE EVENT

A noise event is defined as any SPAD avalanche that is not triggered by a photon emitted from the VCSEL. Noise events come from artificial light or natural light in two different light paths, which are:

- emitted photons from a noise source to the target, and the target reflects photons to the sensor lens;
- or direct emission of photons to the sensor lens.

Additionally, dark counts can trigger SPAD avalanches, but in practice the dark count rate (DCR) is significantly smaller than the noise rate produced by background ambient light. In this model, a background noise of 14k lux halogen light is applied in the simulation (see table 3.2). And we only consider noise counts that are reflected from the target surface onto the sensor lens (see Fig. 3.2).

When the noise photons are first projected to the target, the total noise power Φ_0 over the whole spectrum can be calculated as

$$\Phi_0 = \frac{E_v A_e}{K_0}, \tag{3.15}$$

where E_v is 14k lux, A_e is the effective area for noise photons and K_0 is the luminous efficacy. For a 14k lux halogen light and the target of 0.2 cm × 0.26 cm, it is assumed that its spectral radiance is similar to a black-body radiator radiance $B_\lambda(T)$ at temperature T=2800K, leading to [6]:

$$B_\lambda(\text{T} = 2800\text{K}) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{2800\lambda k}} - 1}, \tag{3.16}$$

where h is the Plank constant, c is the light speed and k is the Boltzmann constant.

As mentioned above, an optical filter is placed between the detector and the lens (see figure 3.2) with a passband (B_0) of 20 nm and a maximum efficiency of 67% (see table 3.2). It is assumed that the filter passband has a Gaussian shape. Its standard deviation (σ_f) is assumed to be 10 nm and its mean value (μ_f) is given by the central wavelength 940 nm. So, its characteristic equation is given by

$$\eta_f(\lambda) = e^{-\frac{1}{2}(\frac{\lambda-\mu_f}{\sigma_f})^2}, \tag{3.17}$$

where λ is the wavelength. The coefficient C_o between the power that can pass the filter band and the total power over the spectrum band ($\Delta\lambda$) can be calculated as

$$C_o = \frac{\int_{\Delta\lambda} \eta_f(\lambda) B_\lambda(T = 2800\text{K}) d\lambda}{\int_{\Delta\lambda} B_\lambda(T = 2800\text{K}) d\lambda}. \tag{3.18}$$

So, the total noise power Φ_n that can pass the filter passband is given by the target reflectivity ρ and equations (3.15) and (3.18):

$$\Phi_n = \Phi_0 \rho C_o. \tag{3.19}$$

The noise rate per time unit (N_r) is calculated by replacing the reflected power and multiplied by laser repetitive frequency into equation (3.14). It is given by

$$N_r = \Phi_n \eta_l \frac{2\lambda}{M\pi hc} \text{PDE}. \tag{3.20}$$

3.3. TIMESTAMP MODEL

The output of the timestamp model stage is a single histogram (**H**), when simulating fixed optical parameters (see Fig. 3.1). Or, multiple histograms when sweeping the optical parameters in the same simulation run. The timestamps of signal and noise events are generated based on the calculation results of equations (3.14) and (3.20).

In a single TCSPC cycle, the number of detected VCSEL photons is a random variable (RV) that follows a Poisson process, which is defined as [7]

$$P(X = k) = \frac{S_p^k}{k!} e^{-S_p}, \tag{3.21}$$

where S_p is the expected number of signal events per TCSPC cycle. We considered that the VCSEL light pulse has a Gaussian shape and dispersion effects are neglected. Therefore, the timestamp of a signal event is represented by an RV with a probability density function (PDF) given by [8]

$$f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t-\mu_s}{\sigma})^2}, \tag{3.22}$$

where σ that is directly related to the VCSEL pulse width. Also, the mean value μ_s is the average TOF for light traveling from the VCSEL to the target and from the target to the D-TOF LIDAR sensor. As the laser and the sensor located at the same surface, μ_s is given by

$$\mu_s = t_0 = 2\frac{d_T}{c}, \tag{3.23}$$

where d_T is the distance between the laser surface and the target surface (see Fig. 3.2).

The noise events are uniformly distributed over time. Their timestamps, which are calculated as the time distance between the start of a TCSPC cycle and the noise event detection, are represented by an RV with exponential PDF. Therefore, the noise event timestamps' PDF is defined by

$$g(t, N_r) = \begin{cases} 1 - e^{-N_r t} & t \geq 0 \\ 0 & t < 0 \end{cases}. \tag{3.24}$$

Algorithm 1 shows the timestamp generation logic for a single TCSPC simulation. It is assumed that 1 SPAD is activated out of M SPADs, and it is connected to 1 single TDC. An output example histogram from the simulation is shown in Fig. 3.3, in which 30,000 TCSPC cycles were simulated. It is worth mentioning that the SPAD deadtime is not utilized in the way we generate timestamps. In each TCSPC cycle, we only record the first timestamp, and wait for the next TCSPC cycle to start. Its repetitive period is much longer than the SPAD deadtime (see table 3.2). Thus, the SPAD deadtime is not taken into consideration in this model.

input : The average number of signal events per TCSPC cycle S_p
input : The noise event rate per second N_r
input : Average TOF μ_s
input : VCSEL pulse width σ
output : TDC code

for i = 1 : TCSPC **do**
    total_signal_events = `poissrnd` (mean=S_p);
    signal_timestamps [i] = `normrnd` (mean=μ_s, sd=σ);
    noise_timestamp = `exprnd` (mean=N_r);
end
all_timestamps = [noise_timestamp, signal_timestamps];
all_timestamps = sort(all_timestamps, decreasing = `FALSE`)
first_timestamp = all_timestamps [1];
tdc_code = `tdc_rounding` (first_timestamp);

Algorithm 1: Timestamp calculation algorithm.

From Fig. 3.3, we can observe a signal peak and the background noise floor. It can be found that the background noise floor on the left side of the signal peak has more counts than that on the right side of the signal peak. An explanation is that the noise events are uniformly distributed over time and the signal events are distributed in Gaussian. And, we compare the first noise event with the first signal event in each TCSPC cycle to record the earlier event in the histogram. For the noise events that are distributed later than the signal event Gaussian shape, they have a lower probability to be the earlier, compared to the noise events that are before the Gaussian shape. Therefore, the histogram has a background noise shape with a high-left and a low- right.

3.4. DISCUSSION

In this chapter, a model for calculating the photon rate and generating timestamp is introduced and simulated. The most important concept, which is the subdivision of the target into pixels, not only avoids the model from making too many assumptions, but dividing the light cover area into point diffuse sources. Moreover, as discussed in section 3.3, signal and noise photons follow a specific type of distribution, which is possible to

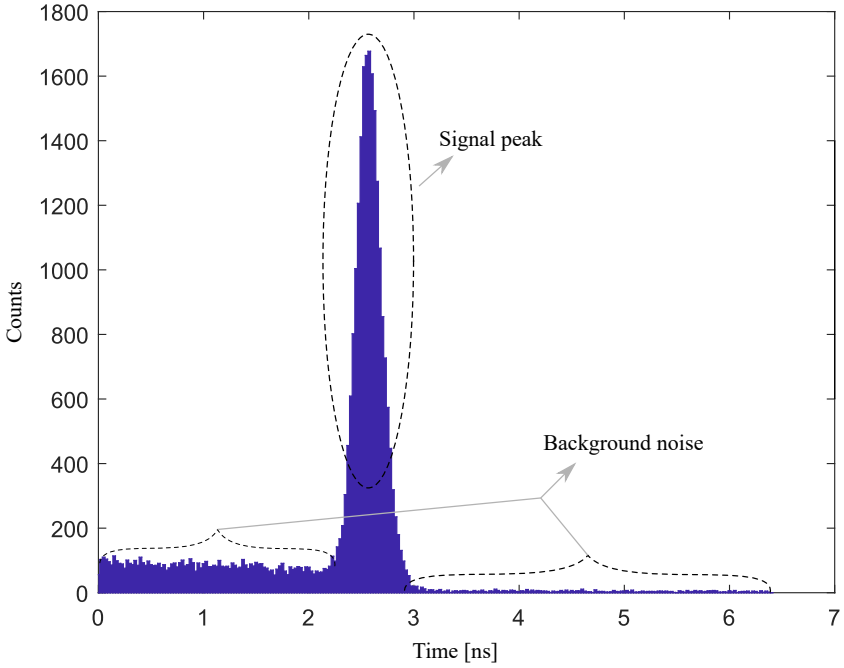


Fig. 3.3: Histogram for D-TOF simulation model.

generate and record timestamps in the simulation. The implemented optical and time-stamp model will serve as a basis for understanding the system trade-offs in chapter 4, in order to design the final system architecture.

BIBLIOGRAPHY

[1] *IEC 60825-1:2014 Safety of laser products – Part 1: Equipment classification and requirements*. Standard. International Electrotechnical Commission, 2014.

[2] Augusto Ronchini Ximenes. "Modular time-of-flight image sensor for light detection and ranging: A digital approach to LIDAR". English. PhD thesis. Delft University of Technology, 2019. DOI: 10.4233/uuid:c434368a-9a67-45de-a66f-f5dc30430e03.

[3] Preethi Padmanabhan, Chao Zhang, and Edoardo Charbon. "Modeling and Analysis of a Direct Time-of-Flight Sensor Architecture for LiDAR Applications". In: *Sensors* 19.24 (2019). ISSN: 1424-8220. DOI: 10.3390/s19245464. URL: https://www.mdpi.com/1424-8220/19/24/5464.

[4] *FB940-10*. (Last accessed: 31/05/2022). Thorlabs, Inc., 2003. URL: https://www.thorlabs.com/thorproduct.cfm?partnumber=FB940-10.

[5] M.D. Adams. "Lidar design, use, and calibration concepts for correct environmental detection". In: *IEEE Transactions on Robotics and Automation* 16.6 (2000), pp. 753–761. DOI: 10.1109/70.897786.

[6] David G Andrews. *An introduction to atmospheric physics*. Cambridge University Press, 2010.

[7] Roy D Yates and David J Goodman. *Probability and stochastic processes: a friendly introduction for electrical and computer engineers*. John Wiley & Sons, 2014.

[8] Gordon Leslie Squires and Gordon Leslie Squires. *Practical physics*. Cambridge university press, 2001.

4

SYSTEM TRADE-OFF ANALYSIS

Mingzhe CHEN

System trade-off analysis is a powerful tool to study parameter dependencies, which helps us find the detailed specifications that meet the system requirements. For example, it enables us to evaluate system linearity and accuracy, showing the performance of the system under different conditions. In this chapter, we perform a system trade-off analysis by using the modeling described in the previous chapter.

4.1. INTRODUCTION

In the histogram obtained from the simulation model, we find that two effects interfere with our ability in obtaining accurate depth information (see Fig. 4.1). First, the level of background noise needs to be low enough to allow the signal peak to be detected. Second, the detected signal peak is shifted with respect to the actual target position, and this effect can be explained by order statistics.

To visualize the impact of background noise, a trade-off analysis can be performed to evaluate the depth range, which is defined as the range of depths that the system can detect. By checking the detection range under different parameter conditions, we can observe the effect of the system parameters in the maximum range. The background noise rate is the main parameter that restricts the depth range. However, other parameters are important, such as VCSEL power, target reflectivity, TDC bin size, and the total number of TCSPC cycles. This is further elaborated in section 4.2.

Furthermore, it is possible to utilize trade-off analysis to evaluate the system depth resolution, which includes the linearity as well. This process is based on extracting features from the timestamp histograms, and it is elaborated as depth resolution analysis in section 4.3.

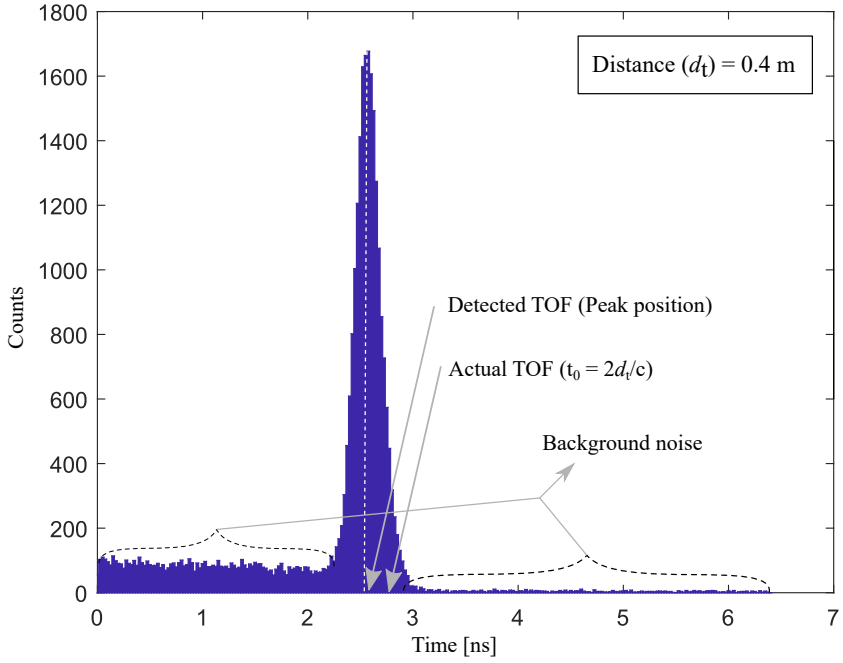


Fig. 4.1: Non-linearities and background noise effects in the output histogram of the 1D-TOF simulation model.

Once we find an optimal parameter selection, the next step is to perform a failure rate analysis. We define failure rate as the inverse of the probability that the system fails, during a certain number of operation cycles under the given specifications. In section 4.4, we perform failure rate analysis to assess the system performance under our optimal parameter selection.

4.2. DEPTH RANGE

The aim of the depth range analysis is to find the maximum distance the system can detect under critical conditions. And, the minimum distance the system can detect is predefined to be 0.1 m (see table 3.2). In the depth range analysis, we do not put special effort to identify the closer distance, but check if the system can work under the minimum distance of 0.1 m. In all the later simulations, if the distance serves as a parameter, it will be swept starting from 0.1 m.

This analysis is performed based on the histograms generated for different parameter conditions, and detecting the depths corresponding to their signal peaks. A parameter sweeping simulation is run, which attempts to find the depth range under such conditions.

For every fixed group of parameters, a histogram can be generated by the simulation

model. By increasing the distance of the target and sweeping the parameters in the same run, we obtain a set of histograms for every target distance point. In addition, we take advantage of a peak detection (PD) algorithm of MATLAB to perform the peak detection. Typically, the PD algorithm can find the peak with respect to the peak's prominence, peak width, etc. These algorithm details are carefully described in section 5.1.

To perform the depth range analysis, the operation steps are listed as follows.

- First, we define a failure condition (FC) as, when PD algorithm is unable to locate a signal peak in 10 out of 100 histograms under the same conditions.
- Next, when FC occurs in the histogram set of one distance point with the fixed parameters, these fixed parameters are considered as the critical conditions. And this distance is considered as the maximum depth range under such critical conditions.

We first classify the parameters into adjustable and non-adjustable. The adjustable parameter is the internal parameter of the system that we can control, and the non-adjustable parameter is the external parameter that depends on the application parameters. Table 4.1 and 4.2 show the adjustable and non-adjustable parameters, respectively. It is worth mentioning that, from the noise event analysis in subsection 3.2.2, we find that the halogen noise level from specs is similar to 100k lux natural light at sea level, which is equal to 0-0.4 W/m^{2}nm solar irradiance at sea level [1]. Thus, we then use solar irradiance (E_e) as the swept parameter in the simulations.

Adjustable parameter	**Description**
P_s	Laser power
TCSPC	Total time-correlated single photon counting cycles
TDC resolution	Average TDC bin size

Table 4.1: Adjustable parameters in the system.

Non-adjustable parameter	**Description**
E_e	Noise (irradiance)
ρ	Target reflectivity

Table 4.2: Non-adjustable parameters in the system.

It is important to mention that the PDP ranges from 1% to 2%. It is a potential adjustable parameter, which is evaluated separately. The modeling analysis in the previous chapter indicates that, changing the PDP has an impact on the noise rate. Therefore, in practice we simulate the depth range within the PDP range under the conditions that may be the worst [1]. The simulation results can be found in Fig. 4.2.

Fig. 4.2 indicates that, when having a high background noise, the increase of PDP decreases the depth range. To be specific, the increase of PDP also increases the noise rate, and it makes the signal peak be difficult to be detected. In order to have a PDP

[1] This 'worst case' is presented in detail in the following paragraphs.

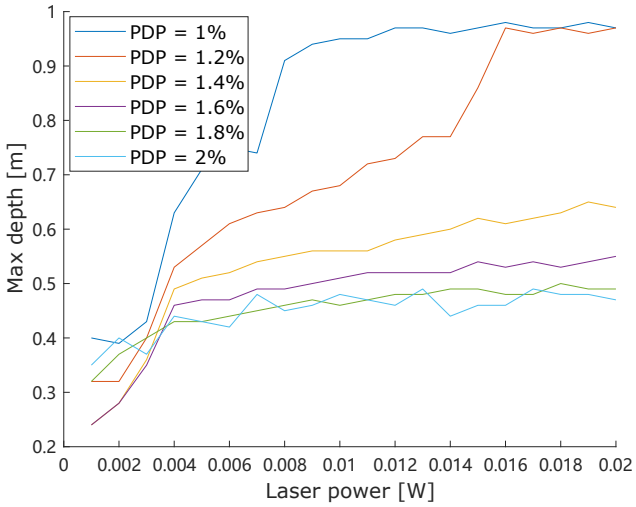


Fig. 4.2: Depth range under different PDP. The simulation parameters are set as: TCSPC = 30,000, TDC bin size = 25 ps, E_e = 0.4 W/m^{2}nm, and ρ = 60%.

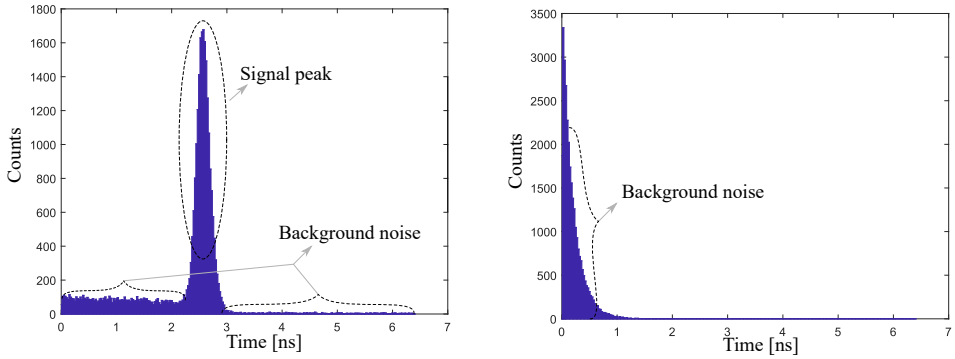
value that can detect at maximum 0.6 m [2], we picked 1.2% as the PDP in all the later simulations.

Besides, in order to have a low noise rate, we activate one SPAD of the array out of 40, which is connected with one TDC, in all simulations[3]. The reason for having one SPAD activated is that, the system will easily fail when more SPADs are activated, when the system only comprises of one TDC. To be specific, when more SPADs are activated and connected to one TDC, the TDC records the first timestamp among all the activated SPADs, showing a 'winner-take-all' behavior. However, due to the uniform distribution of background noise events, the TDC will more likely to be triggered by a noise timestamp, and the final histogram does not contain the signal peak (see Fig. 4.3). So, effectively the background noise rate at the TDC input is reduced.

We consider the laser power (P_s) is the easiest controllable parameter in the system. Thus, each time we simulate the depth range, we also sweep the laser power up to its maximum value, in order to find an optimal laser power at different conditions. The laser power is an important parameter that determines the total power consumption of the LIDAR system. Fig. 4.4 shows the maximum depth under different parameter conditions. And, the observation can be summarized as follows:

- The depth range increases with the increase of P_s.
- The system fails when the total number of TCSPC cycles is lower than 500. When the system starts working, the depth range increases with the increase of the total

[2] In table 3.2, we define the detection range is 0.1 - 0.6 m.
[3] It is assumed that this TDC can only record the first photon timestamp.



(a) (b)

Fig. 4.3: The simulated performance with activating different number of SPADs, connected with one single TDC. (a) 1 activated SPAD; (b) 40 activated SPADs.

TCSPC cycles [4].

- The depth range increases with the increase of TDC bin size.
- The depth range decreases with the increase of E_e.
- The depth range first increases with the increase of ρ, but becomes almost unchanged when ρ reaches its maximum.

Moreover, we perform further analyses, in order to explain the potential questions from the observations.

- **TCSPC:** We want to know what happened when the total TCSPC cycle is lower than 2000. A further fine-sweep is made, by sweeping TCSPC cycle from 500 to 2000, as shown in Fig. 4.5(a). In this improved fine-sweep, we can observe that the depth range increases with the increase of TCSPC cycles.
- **TDC bin size:** From our observation in Fig. 4.4(b), the larger the TDC bin size is, the higher depth range is. Essentially, the TDC bin size defines the histogram bin size. And, with the same number of bin, the TDC full scale range will increase accordingly. Thus, the increase of TDC bin size increases the depth range. To explain this, we first give a plot as an example, as shown in Fig. 4.6. A clear observation is that, at the same distance point, more counts locate in the signal peak region with larger TDC bin size, which makes the signal peak be easily detected.
- **E_e & ρ:** We present the depth range simulation with both ρ = 8% and ρ = 60%, showing the target reflectivity and background noise irradiance are highly dependent parameters when determining the depth range (see Fig. 4.5(b)). It can be

[4] The depth range for TCSPC = 500 remains at 0.1 m. As 0.1 m is the lowest value of the swept distance value, we consider the system fails under this condition.

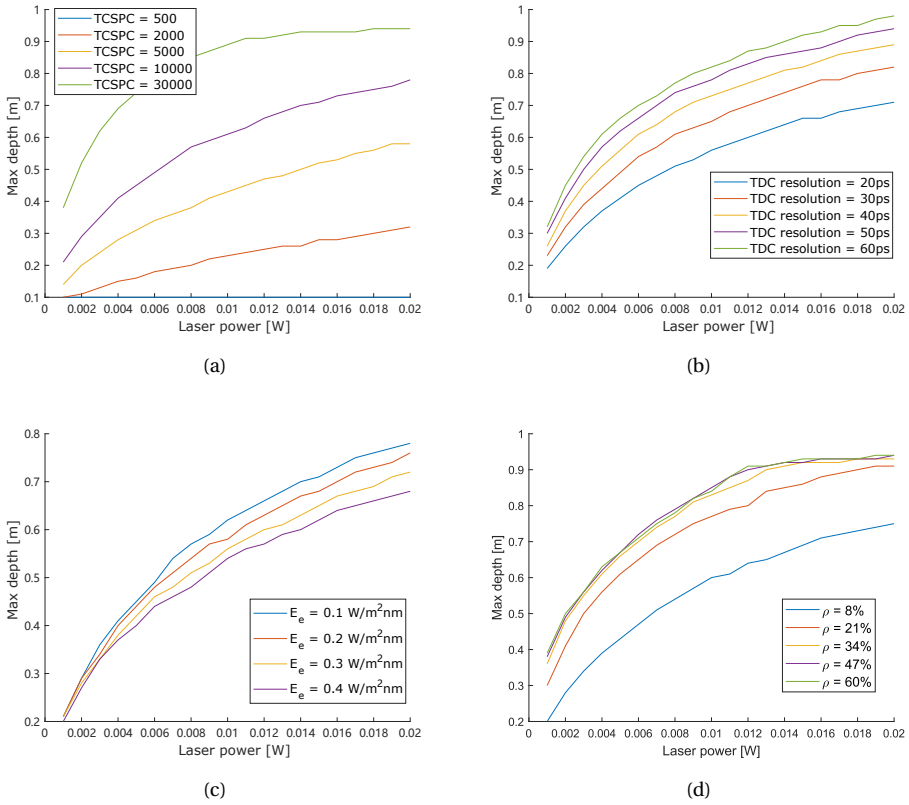
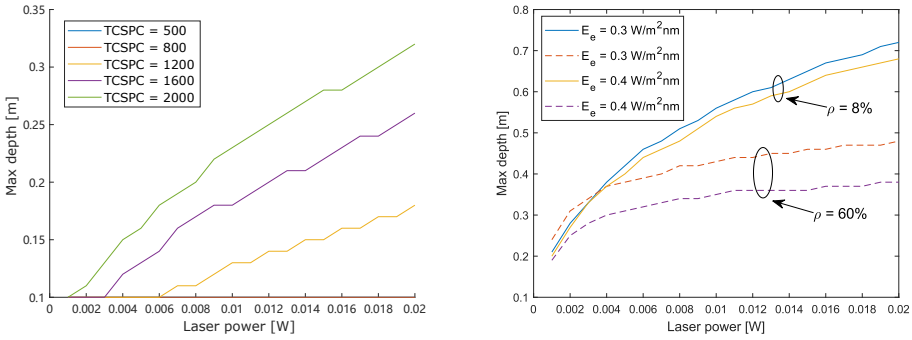


Fig. 4.4: Depth range under adjustable parameter conditions: (a) Depth range vs. Total TCSPC cycles; (b) Depth range vs. TDC bin size. Depth range under non-adjustable parameter conditions: (c) Depth range vs. Noise irradiance; (d) Depth range vs. Target reflectivity. Laser power is the x-axis of the simulation, which is used to offer the curve under initial condition. When the corresponding parameters are not swept, they are set as: PDP = 1.2%, TCSPC = 30,000, TDC bin size = 25 ps, E_e = 0.1 W/m^{2}nm, and ρ = 8%.

found that the 60% ρ has a higher depth range at the beginning, and becomes saturated when increasing the P_s. The 8% ρ first has a lower depth range, but its depth range continues to increase. Crossing point can be found for each two curves with the same background noise and different ρ, in which the simulations under both ρ give the same depth range. The explanation to this observation that in the curve with lower ρ, the depth range is restricted by the laser power, as the noise does not saturate the model; in the curve with higher ρ, the depth range is restricted by the noise showing its saturation. Moreover, as the increase of ρ can both increase the number of signal and noise photons, the impacts from both photons contribute together to the depth range. Although the depth range changes the trend with the increase of ρ, we only take its 'worst case' into consideration, in which the simulation can cover all possible parameter conditions.



(a) (b)

Fig. 4.5: Improved sweep: (a) a fine-sweep for depth range under total TCSPC cycles from 500-2000; (b) depth range sweep at $\rho = 8\%$ and 60% with respect to Fig. 4.4(c). When the corresponding parameters are not swept, they are set as: TCSPC = 30,000, TDC bin size = 25 ps, $E_e = 0.1$ W/m^{2}nm, and $\rho = 8\%$.

From the simulation result of depth range, we are able to find the maximum range of the system. We first set the worst-case for non-adjustable parameters, which leads to highest background noise and highest reflectivity (see Fig. 4.4). Then, the maximum depth range can be found by sweeping other adjustable parameters. It can be found that the maximum detection range is around 0.3 m under worst-case conditions (see Fig. 4.7). However, it is essential to mention that, this worst-case has reached the limits of the utilized PD algorithm. Although there is indeed a protruding signal peak, the algorithm can still miss that peak. This is further elaborated in section 5.1.

4.3. DEPTH RESOLUTION

Depth resolution analysis is defined as the method of studying the depth estimation mean-square-error (MSE) of the system. In this section, we also use the PD algorithm from section 5.1 as our estimation method to estimate the signal peak position. Also, the peak shift effect is quantified, in order to help us understand the relation between non-linearities and the system parameters.

The peak shift effect can be explained by the bias (b) shift of the main peak. Also, the detections of the signal peak show a dispersion that can be explained by the standard deviation (σ). Thus, we measure the MSE of our estimation method, which associates b and σ together. This relation is given by [2]

$$\mathrm{MSE} = \sigma^2 + b^2. \tag{4.1}$$

Therefore, the peak shift effect is evaluated through MSE of our estimation method. We also calculate the sample MSE ($\hat{\mathrm{MSE}}$) as

$$\hat{\mathrm{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(\hat{\mathrm{D}}_i - \mathrm{D}_0)^2, \tag{4.2}$$

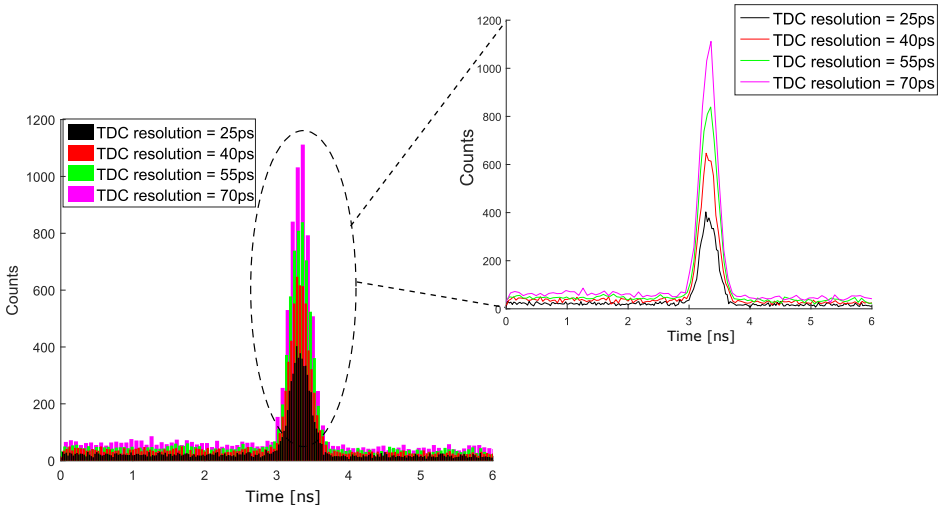


Fig. 4.6: Histogram difference while sweeping TDC bin size. The zoom-in line plot uses the bin center as the x coordinate and the bin counts as the y coordinate.

where n is the number of measurements, $\hat{D}_i$ is the detected TOF in the i_{th} prediction, and D_0 is the actual depth.

However, as a scale-dependent measure [3], $\hat{MSE}$ is in the unit of [m^2] in our estimator. For a better understanding and observation, we use $\sqrt{\hat{MSE}}$ (in the unit of [m]) instead, in order to show the depth difference directly.

Hence, we define $\sqrt{\hat{MSE}}$, σ and b as the performance criteria of the depth resolution analysis. The peak location ($\hat{D}_i$) for the histograms from the previous simulations are then obtained, which are used for calculating the $\sqrt{\hat{MSE}}$. σ and b of the histograms are calculated from 1,000 histograms under same parameter condition. By sweeping the parameters in the run, the relation between the depth resolution and the adjustable parameters are visualized (see Fig. 4.8). Also, we simulate the depth resolution performance under non-adjustable parameters (see Fig. 4.9).

There are three points that need to be explained for readers' better understanding:

- In the depth resolution simulation, we first simulate the depth resolution while sweeping the total TCSPC cycles. We find that the $\sqrt{\hat{MSE}}$ remains unchanged with the increase of total TCSPC cycles (see Fig. 4.8(a)). The b is unchanged and the σ increases by a very small order of magnitude (see Fig. 4.8(b)). Moreover, in Fig. 4.4(a), at 7.36 mW laser power, the curve with TCSPC = 10,000 can reach the detection range given by the specs. Therefore, in order to present the depth resolution

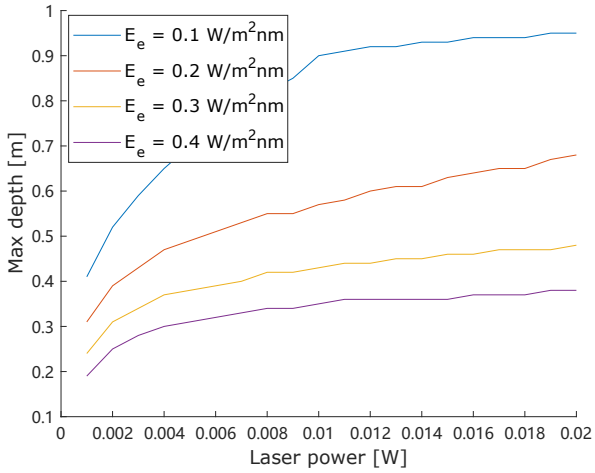


Fig. 4.7: Depth range under worst-case conditions. The non-swept parameters are set as: PDP = 1.2%, TCSPC = 30,000, TDC bin size = 25 ps, and ρ = 60%.

while reducing the simulation time, we use TCSPC = 10,000 to simulate the depth resolution with other parameters.

- The depth resolution vs. TDC bin size simulation shows that, the lower TDC bin size is, the more stable σ is (see Fig. 4.8(d)). An explanation to this is that, when the target depth locates near the boundary of the bin, the obtained peak value will jump between the adjacent bins, showing a large fluctuation of σ. However, when the target depth is located in the center of the TDC bin, the σ fluctuations are smaller. This effect gets more obvious when choosing large bin sizes (see Fig. 4.8(d)).

- The main factor that increases the $\sqrt{\hat{\text{MSE}}}$ is the b and is not the σ. The change of the b is explained by order statistics because the first photon time PDF is shifted with respect to the unsorted photons' time PDF [4, 5].

4.4. FAILURE RATE

We consider that the PD algorithm is the simplest method, which extracts signal information from the histogram. Hence, we use it as a reference method for our failure rate analysis (FRA). Essentially, FC occurs when the parameters are set near the critical conditions. Therefore, we propose the FRA, which is used to evaluate whether a histogram carries distinguishable information in the detection range under the critical conditions.

To select the parameter sweep space in FRA, we propose the following criteria:

- the non-adjustable parameters are swept linearly within their range, defined as parameter sweeps. For example, the E_e is swept from 0 to 0.4 W/m^{2}nm and the ρ

Distance checkpoint [m]	Failure rate (from 30,000 simulations)
0.1	0
0.2	0
0.3	0
0.4	7
0.5	332
0.6	2584

Table 4.3: Failure rate at six critical distances.

is swept from 8% to 60%. In total we simulate 300 parameter values and each of them is simulated for 100 times. Thus, at every distance, 30,000 simulations are performed.

In table 4.3, the FRA simulation result is presented. The failure rate percentage represents how many times the PD algorithm failed within the 30,000 simulations.

It is given that the model with PD algorithm has a failure rate within 2584 fails in all distance values. In particular, we hardly detected FRA occurrence with 30,000 simulations when the distance is within 0.4m.

The results in table 4.3 can also explain the observation is Fig. 4.7, in which the maximum depth range under worse-case conditions only reaches 0.4 m. This is because the maximum depth range simulation is performed under $\rho = 60\%$ and $E_e = 0.4\text{W/m}^2\text{nm}$, and this parameter choice just locates in the region where the PD algorithm could fail.

4.5. DISCUSSION

In this chapter, we performed a system trade-off analysis, to investigate the effects in the TOF histogram. The trade-off analysis contains three topics, which are depth range, depth resolution, and failure rate analysis. The depth range analysis presented the system detection range under different parameter conditions. The depth resolution analysis simulated the MSE of the system, and visualized the peak shift effect of the signal peak. Moreover, the failure rate analysis was performed, in which we evaluate the distinguishable information's existence under critical conditions.

From the system trade-off analysis, we observe the effects in the TOF histogram. Therefore, the next step is to design the algorithms that can compensate those effects and identify the target depth.

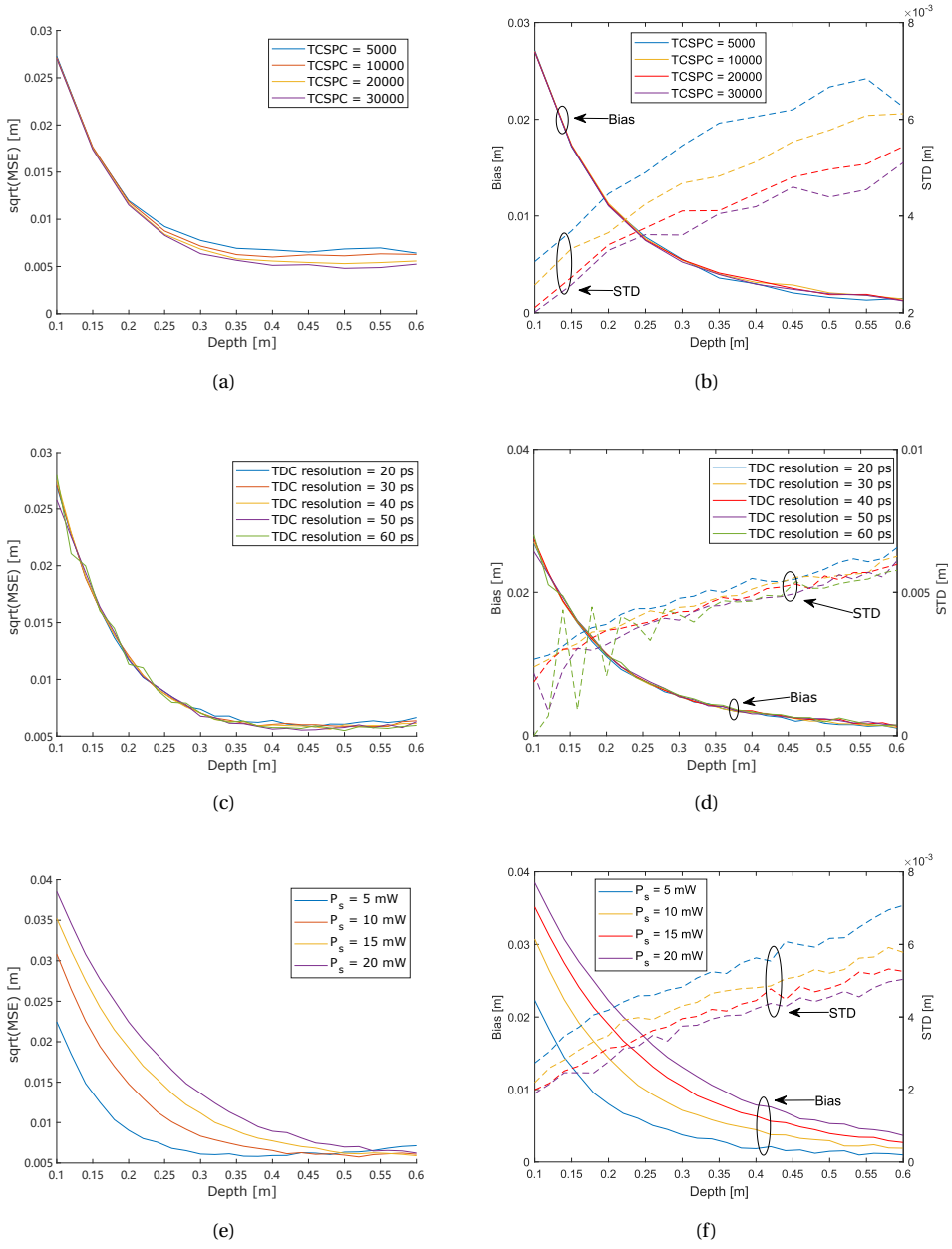


Fig. 4.8: Depth resolution under adjustable parameters: (a) $\sqrt{\widehat{\mathrm{MSE}}}$ vs. Total TCSPC cycles, (b) b, σ vs. Total TCSPC cycles; (c) $\sqrt{\widehat{\mathrm{MSE}}}$ vs. TDC bin size, (d) b, σ vs. TDC bin size; (e) $\sqrt{\widehat{\mathrm{MSE}}}$ vs. Laser power, (f) b, σ vs. Laser power. When the corresponding parameters are not swept, they are set as: PDP = 1.2%, P_s = 7.36 mW, TCSPC = 10,000, TDC bin size = 25 ps, E_e = 0.1 W/m^{2}nm, and ρ = 8%.

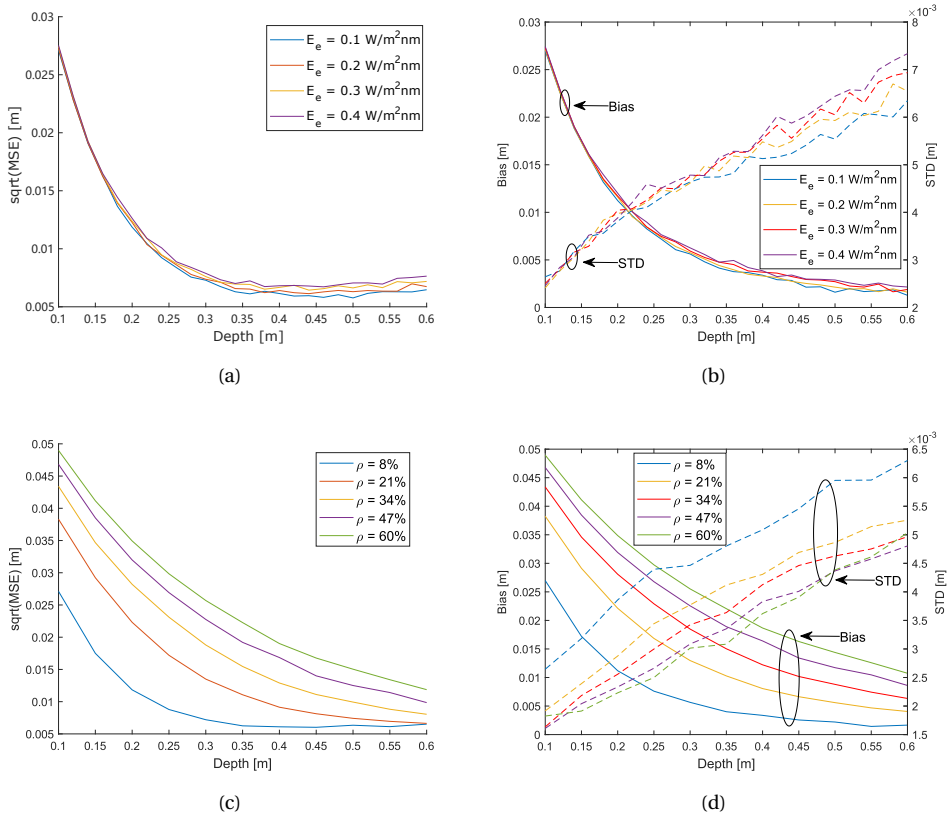


Fig. 4.9: Depth resolution under non-adjustable parameters: (a) $\sqrt{\hat{\mathrm{MSE}}}$ vs. Noise irradiance, (b) b, σ vs. Noise irradiance; (c) $\sqrt{\hat{\mathrm{MSE}}}$ vs. Target reflectivity, (d) b, σ vs. Target reflectivity. When the corresponding parameters are not swept, they are set as: PDP = 1.2%, P_s = 7.36 mW, TCSPC = 10,000, TDC bin size = 25 ps, E_e = 0.1 W/m^{2}nm, and ρ = 8%.

BIBLIOGRAPHY

[1] Richard E. Bird and Carol Riordan. "Simple Solar Spectral Model for Direct and Diffuse Irradiance on Horizontal and Tilted Planes at the Earth's Surface for Cloudless Atmospheres". In: *Journal of Applied Meteorology and Climatology* 25.1 (1986), pp. 87–97. DOI: 10.1175/1520-0450(1986)025<0087:SSSMFD>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/apme/25/1/1520-0450_1986_025_0087_sssmfd_2_0_co_2.xml.

[2] Wikipedia. *Mean squared error*. [Online; last accessed 20-June-2022]. 2022. URL: https://en.wikipedia.org/wiki/Mean_squared_error.

[3] Rob J. Hyndman and Anne B. Koehler. "Another look at measures of forecast accuracy". In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000239.

[4] Matthew W. Fishburn and Edoardo Charbon. "System Tradeoffs in Gamma-Ray Detection Utilizing SPAD Arrays and Scintillators". In: *IEEE Transactions on Nuclear Science* 57.5 (2010), pp. 2549–2557. DOI: 10.1109/TNS.2010.2064788.

[5] Esteban Venialgo et al. "Time estimation with multichannel digital silicon photomultipliers". In: *Physics in Medicine and Biology* 60.6 (Mar. 2015), pp. 2435–2452. DOI: 10.1088/0031-9155/60/6/2435. URL: https://doi.org/10.1088/0031-9155/60/6/2435.

5

ESTIMATION ALGORITHM DESIGN

Mingzhe CHEN

The main purpose of the 1D-TOF system is to estimate the distance from the sensor to the target. Histograms obtained from modeling contain distinguishable information, in which we can derive the travel time of the VCSEL photons. However, the analysis in the previous chapter shows the non-linearities in the depth estimation when using the PD algorithm. The peak shift effect and the background noise prevent us from obtaining accurate depth from the distinguishable information in the histogram. Therefore, histogram processing algorithms are required to correct these non-linearities and derive the actual depth from histograms. In this chapter, the PD algorithm mentioned in section 4.2 is first discussed. A noise rejection algorithm is then introduced. Finally, we propose the use of artificial neural networks (ANNs) as a robust and unbiased depth estimator.

5.1. PEAK DETECTION ALGORITHM

First, we use the PD algorithm to seek the peak in a histogram as our performance reference. The feature in the raw histogram that we are most interested in is the signal timestamping peak (see Fig. 5.1). It represents the distribution of the detected photons from the VCSEL. According to the analysis in section 4.3, the shape of the signal peak has a left-skewed shape originating from Gaussian distribution. Therefore, the shape has its own unique dominant peak. In this section, we further evaluate the performance of the PD algorithm to detect the signal dominant peak in the histogram. In addition, a correction of depth is performed to compensate peak shift effect.

5.1.1. ALGORITHM DESCRIPTION

The PD algorithm consists of three stages. The first stage fits a smoothing curve using bin center and counts. The second stage calculates the main peak of the smoothed curve.

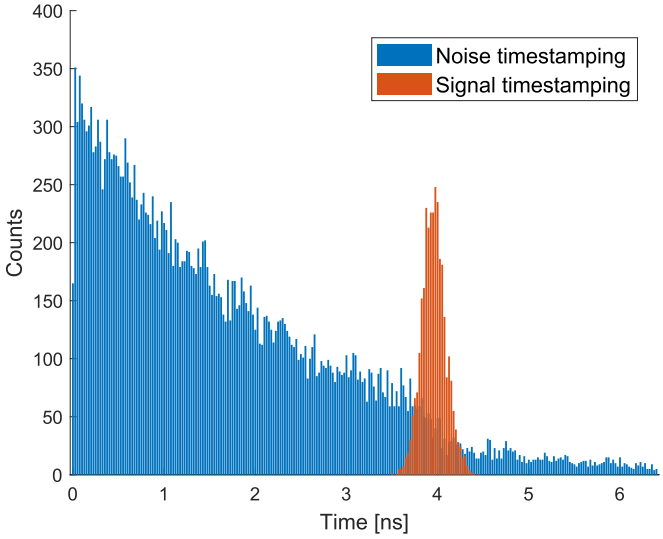


Fig. 5.1: TOF histogram generated from modeling. The adjustable parameters use the value in table 3.2, and the non-adjustable parameters use the random value in their available value range ($E_e \in 0$ - 0.4 W/m^{2}nm, and $\rho \in 8\%$ - 60%).

Last, a compensation equation is utilized in the third stage, in which the compensation of the peak shift effect is performed.

In the first stage, we perform spline interpolation with not-a-knot end conditions, in order to filter the histogram poisson noise (see Fig. 5.1). In MATLAB, it is known as 'spline' attribute of function *interp1*. The code is given by

Listing 5.1: Interpolation MATLAB code.

```
X_N = linspace(min(centers),max(centers));
Y_N = interp1(centers,counts,X_N,'spline');
```

where counts and centers are the old coordinates from the histogram bin property. X_N, Y_N are the new coordinates after interpolation [1].

Attribute	Value	Description
Minpeakprominence	0.003∗TCSPC	Prominence value
MaxPeakWidth	FWHM_LASER	Maximum peak width
Annotate	extents	Add annotation
WidthReference	halfheight	Measure the width referring to half-height

Table 5.1: Attributes of *findpeaks* function.

In the second stage, we utilize *findpeaks* function to the updated coordinates. This function is included in the *Signal Processing Toolbox* of MATLAB, which can locate the

peak of the curve [2]. The attribute 'Minpeakprominence' is used, in order to avoid misdetections originating from the Poisson noise in the counts of the histogram. The attribute 'MaxPeakWidth' is used to avoid that the function recognizes noise-exponential slope as a peak. It forces the function to discard the peak that is wider than the VCSEL full-width-half-maximum. Other attributes including 'Annotate' and 'WidthReference' are put in the code, to store and display the properties of the peak. A general description of these attributes can be found in table 5.1. It is worth noting that 0.003∗TCSPC in the table 5.1 is the actual value for the function to identify the prominence. This value is selected by slowly increasing the prominence, until the statistical factor shows no impact on the signal peak. This means, the function rejects the peak whose value is less than 0.003∗TCSPC with respect to its neighboring coordinates.

And the code for finding the histogram peak can be found as:

Listing 5.2: Peak finding output arguments and attribute settings MATLAB code.

```
[pks,locs,w,p] = findpeaks(Y_N,X_N,'MinPeakProminence' ...
                 0.003TCSPC,'Annotate','extents' ...
                 'WidthReference','halfheight'...
                 'MaxPeakWidth',FWHM_LASER);
```

A description of the output arguments is listed here, in which *pks* is the height of the peak, *locs* is the time-of-flight of the peak, w is the width at half height and p is the prominence.

In the third stage, we want to obtain the depth by using the output arguments and system features. We first convert the TOF represented by ($locs$) into depth ($locs_{\mathrm{d}}$).

$$locs_{\mathrm{d}} = \frac{c}{2} locs. \tag{5.1}$$

Next, we propose a compensating equation to compensate the effect of peak shift to get the estimated depth D'. Based on the output argument $locs$ and w, a linear compensation is made as follows:

$$D' = K_{\mathrm{d}} locs_{\mathrm{d}} + B_{\mathrm{d}}, \tag{5.2}$$

where K_{d} and B_{d} are coefficients of the linear compensation. And in practice, these two coefficients are calculated based on fitting the typical obtained depth with the target depth. The MATLAB code is given by

Listing 5.3: Calculation of linear compensation coefficients MATLAB code.

```
X = [locs1,locs2,..,locsN];
Y = [D1,D2,..,DN];
P = polyfit(X,Y,1);
```

For readers' convenience, we use ellipses (..) and 'N' suffixes in the code. However, they should be replaced by actual values in the scripts. After applying the calculation, K_{d} and B_{d} are obtained and stored in the variable P. We obtain that K_{d} is 0.9496, and B_{d} is 0.0338.

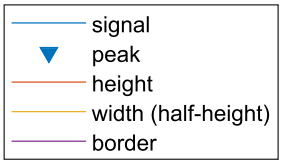


Fig. 5.2: The legend of the features of the *findpeaks* function.

5.1.2. SIMULATION

The simulation aims to present the estimation results of this algorithm. We start with introducing the simulation conditions, and summarize the simulation results.

First of all, from the FRA in section 4.4, we observe that the PD algorithm could fail to detect the peak under conditions near the 'worst case'. Therefore, we define two types of detection, namely 'successful detection' and 'failed detection'. The former one means that the algorithm identifies the signal peak from the histogram, and the latter one means that the algorithm is not able to locate the signal peak. For example, as shown in Fig. 5.3, there is no peak found by the algorithm, although a protuberance appears around 4 ns. To ensure the same sample size in the further performance evaluation, in this subsection, the presented simulation results are based on the 'successful detection', only.

Fig. 5.4 summarizes the simulation results for applying *findpeaks* function and the compensation equation to the cells (a) to (f). The histograms in those cells are generated from the model simulation at distance 0.1, 0.2,...,0.6 m, with random background noise irradiance ranging from 0 to 0.4 W/m^{2}nm. The second column shows the features of *findpeaks* function. And, the legend of the features in the second column is given in Fig. 5.2. The third column shows the numeric results after applying depth compensation formula. It includes the actual depth in the simulation, and the error between the actual depth and the estimated depth from the algorithm. For each cell, the obtained depth is calculated based on 1000 'successful detection' simulations. The obtained depth is expressed in a form of 2 standard deviation formula, and it is given by

$$\text{Obtained depth [m]} = \mu \pm 2\sigma, \tag{5.3}$$

where μ is the mean value and $\sigma$is the standard deviation of 1000 'successful detection' simulations.

5.1.3. PERFORMANCE EVALUATION

From the simulation results, we are able to evaluate the performance of the peak detection algorithm. In this subsection, the advantage and disadvantage of this algorithm are discussed.

The PD algorithm is an intuitive algorithm, which can extract signal statistical features and find the signal peak. Its accuracy is evaluated by comparing the estimated error with the required error range in the system specs (see table 3.2). The error comparison is shown in table 5.2, which is converted by subtracting the obtained depth from the actual depth in the third column of Fig. 5.4. It can be found that most of the estima-

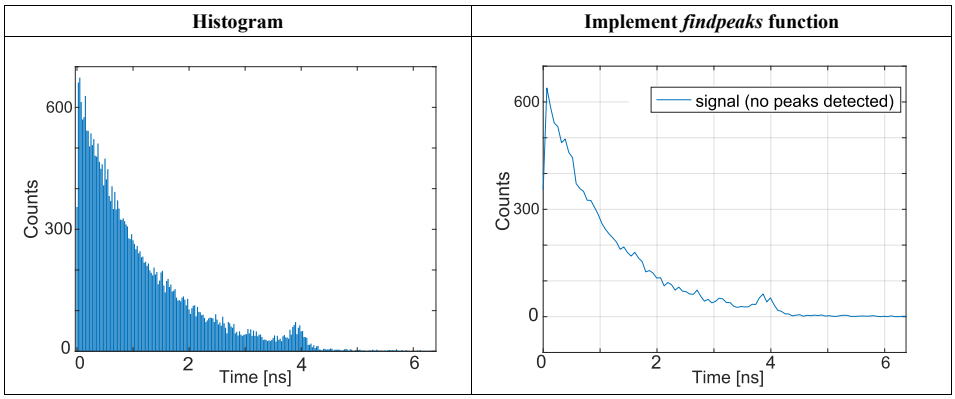


Fig. 5.3: Failure case of the *findpeaks* function. The simulation conditions are set as: TCSPC = 30,000, TDC LSB = 25 ps, P_s = 7.36 mW, E_e = 0.4 W/m^{2}nm, and ρ = 60%.

tions have their error locating within the required error range. We can also notice that, the precision of PD algorithm is around ±0.01 m. This can become very critical at close distance point, as the close distance point requires a lower absolute error.

Distance checkpoint [m]	Requested Error [m]	Estimated Error [m]
0.1	± 0.010	-0.0057± 0.0120
0.2	± 0.010	0.0022± 0.0102
0.3	± 0.015	0.0039± 0.0096
0.4	± 0.020	0.0036± 0.0100
0.5	± 0.025	0.0007± 0.0100
0.6	± 0.030	-0.0023± 0.0095

Table 5.2: Performance of PD algorithm.

However, the drawback of this algorithm is, that it is very rigid in checking the relative prominence height of the peaks. The higher the background noise is, the higher the exponential counts are. As the total TCSPC cycle is constant, it causes that the signal peak has less counts. And this can cause the algorithm reject the main signal peak as well. For example, in Fig. 5.3, the algorithm tells us that 'no peaks detected'. This is the cause to 'failed detection' in the subsection 5.1.2, which also contributes to the failure rate simulation results in table 4.3.

5.2. NOISE REJECTION ALGORITHM

We propose a background noise rejection method called background 'subtraction' in this section. Essentially, as indicated in section 3.3, the detected photons from any noise sources show a pseudo exponential distribution over time. In a histogram, we can observe noise timestamping distribution among the bins, shown as an exponential slope (see Fig. 5.1). To lower the noise level, previous studies focus on rejecting noise

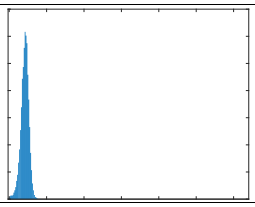
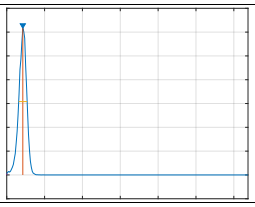
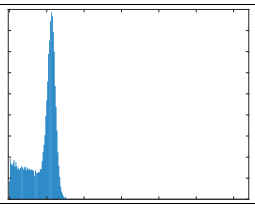
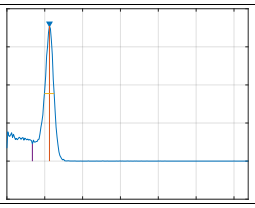
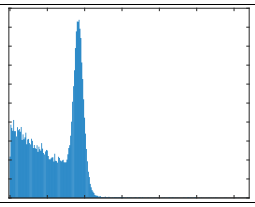
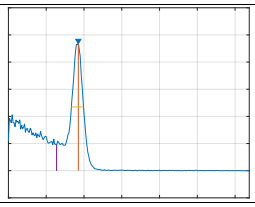
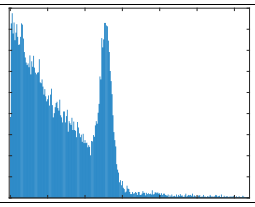
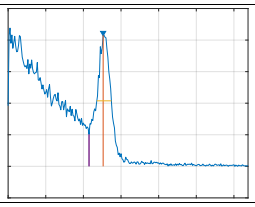
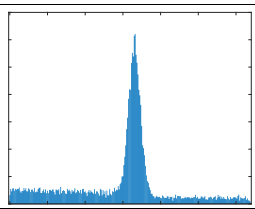
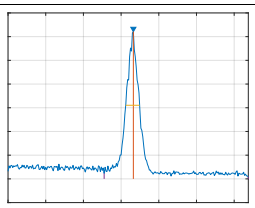
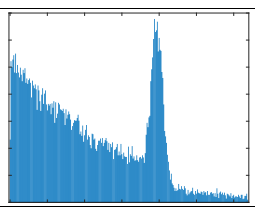
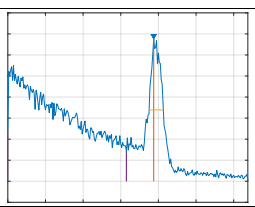
Histogram	Implement *findpeaks* function	Apply depth compensation	
(a) 		(From 1000 simulations)	
		Target depth [m]	0.1000
		Obtained depth [m]	0.0943±0.0120
(b) 		(From 1000 simulations)	
		Target depth [m]	0.2000
		Obtained depth [m]	0.2022±0.0102
(c) 		(From 1000 simulations)	
		Target depth [m]	0.300
		Obtained depth [m]	0.3039±0.0096
(d) 		(From 1000 simulations)	
		Target depth [m]	0.400
		Obtained depth [m]	0.4036±0.0100
(e) 		(From 1000 simulations)	
		Target depth [m]	0.500
		Obtained depth [m]	0.5007±0.0100
(f) 		(From 1000 simulations)	
		Target depth [m]	0.600
		Obtained depth [m]	0.5977±0.0095

Fig. 5.4: Peak detection algorithm simulation. Each cell contains the raw histogram, the properties derived from *findpeaks* function, the actual depth and the obtained depth. The obtained depth is presented as $\mu \pm 2\sigma$, calculated from 1000 successful simulations. The adjustable parameters are: TCSPC = 30,000, TDC LSB = 25 ps, P_s = 7.36 mW. In the first and second column, the non-adjustable parameters are random values that fit the specs to plot examples. In the third column, the non-adjustable parameters are swept linearly, until 1,000 'successful detections' are obtained, in which $E_e \in$ 0-0.4 W/m^{2}nm and $\rho \in$ 8-60%.

counts before the timestamps are recorded into the histogram. Methods such as 'coincidence detections', 'spatio-temporal modulation' were previously developed, which analyze temporal and spatial correlation between nearby pixels, to decide whether a timestamp is kept or discarded [3–5]. However, those methods are performed by grouping multiple pixels and adding extra digital circuits to evaluate the behavior of the grouped pixels. A disadvantage is that it adds extra complexity to the system. So, we propose an algorithm to filter the noise after a histogram is generated.

5.2.1. ALGORITHM DESCRIPTION

The proposed subtraction algorithm can reject the noise based on two continuous histograms. The former one samples the histogram of counts when the VCSEL is turned off; the latter one samples the normal histogram with VCSEL on. By subtracting the counts at the same bin location in two histograms, a new histogram can be made. In the new histogram, the background noise is filtered out, showing a distinct signal peak.

Algorithm 2 shows the logic for subtraction algorithm for two continuous histograms, in which one output histogram $\mathbf{H_o}$ is obtained.

input : The average number of signal events per TCSPC cycle S_p
input : The noise event rate per second N_r
input : Average TOF μ_s
input : VCSEL pulse width σ
output : $\mathbf{H_o}$

$\mathbf{H_1}$:
for i = 1 : TCSPC **do**
  noise_timestamp = `exprnd` (mean=N_r);
end
all_timestamps = sort(noise_timestamp, decreasing = `FALSE`)
first_timestamp = all_timestamps [1];
tdc_code = `tdc_rounding` (first_timestamp);
$\mathbf{H_1}$ = `hist` (tdc_code);

$\mathbf{H_2}$:
for i = (TCSPC + 1) : (2 ∗ TCSPC) **do**
  total_signal_events = `poissrnd` (mean=S_p);
  signal_timestamps [i] = `normrnd` (mean=μ_s, sd=σ);
  noise_timestamp = `exprnd` (mean=N_r);
end
all_timestamps = [noise_timestamp, signal_timestamps];
all_timestamps = sort(all_timestamps, decreasing = `FALSE`)
first_timestamp = all_timestamps [1];
tdc_code = `tdc_rounding` (first_timestamp);
$\mathbf{H_2}$ = `hist` (tdc_code);

$\mathbf{H_o}$:
$\mathbf{H_o}$ = $\mathbf{H_2}$-$\mathbf{H_1}$; %Subtract on each identical bin

Algorithm 2: Subtraction algorithm.

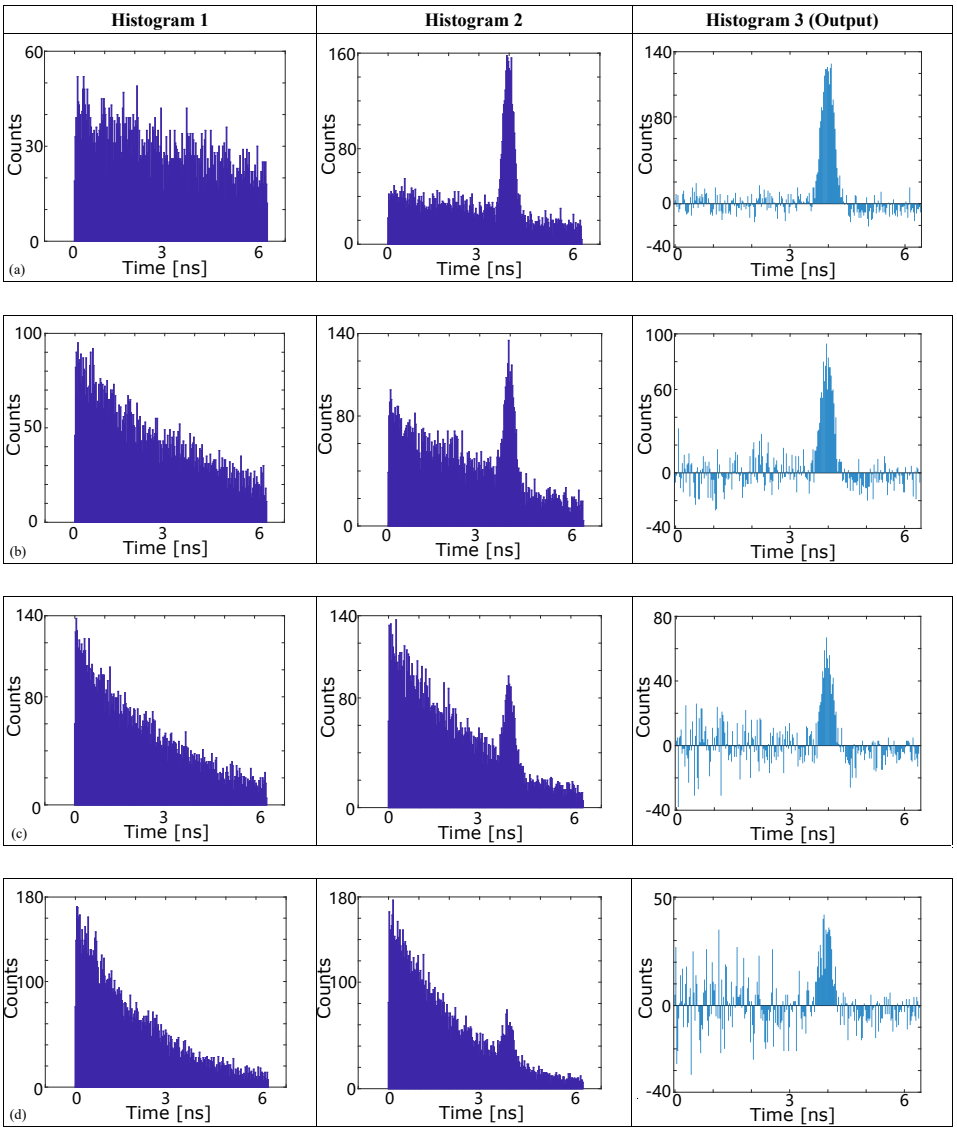


Fig. 5.5: Subtraction algorithm. The simulation parameters are: target distance is 0.6 m, P_s = 7.36 mW, ρ = 60%, TCSPC for **Histogram 1** and **Histogram 2** is 15,000 each, TDC LSB = 25 ps. The background noise is swept from 0.1 to 0.4 W/m^{2}nm with an increment of 0.1 W/m^{2}nm, which corresponds to cell (a) to (d) one by one. **Histogram 1** in the first column is obtained by the modeling simulation with noise only, while **Histogram 2** is simulated with both signal and noise. **Histogram 3** is the output histogram after subtracting **Histogram 1** from **Histogram 2**.

5.2.2. SIMULATION

Fig. 5.5 shows the performance of subtraction algorithm. In the figure, cell (a), (b), (c) and (d) are simulated with incremental background noise conditions, ranging from 0.1 to 0.4 W/m^{2}nm with a step of 0.1 W/m^{2}nm. The first column includes the histograms with only background noise and the second column includes the histograms when turning on the VCSEL. By subtracting the previous histogram (**Histogram 1**) from the latter one (**Histogram 2**), the output histogram (**Histogram 3**) is then obtained.

As we can observe, the noise floor is filtered out to a large extent. Even in the high-noise simulation (cell (d)), the statistics to the noise counts results in a substantial reduction compared to the signal peaks, and the distribution of the noise is even more sparse. It is worth mentioning that the negative count value appears in the **Histogram 3**.

5.2.3. PERFORMANCE EVALUATION

The background subtraction algorithm only serves to provide higher quality histograms, which means it cannot work alone for depth estimation. Therefore, in this subsection, we only present the advantage and disadvantage of this algorithm.

The validity is supported by the theoretical principle. The principle of this algorithm is that two continuous histograms often share the same external conditions, so that the statistical conditions for both histogram background noise are the same. Therefore, the count value at each identical bin of two histograms has the same statistical properties, so that they can be subtracted directly. However, there is a clear bias of negative counts on the right side of the peak in the output histograms (see Fig. 5.5).

The advantage is that this algorithm can assist peak locating algorithm to locate the signal peak much easier. For example, the PD algorithm presented in section 5.1. However, the drawback of this algorithm is the frame rate that is reduced by half. This is because we utilize one more histogram to compute, which takes another time of detection cycle. Of course, we can make a trade-off between the quality of the noise histogram and the total frame rate by reducing the number of TCSPC cycles required to build the noise histogram. But the frame rate will be always reduced to some extent.

5.3. ARTIFICIAL NEURAL NETWORK

Typically, the combination of multiple algorithms leads to an increase in the complexity of the system. For example, the analyses in the previous sections showed the goals, including finding the signal peak and filtering the background noise, require individual approaches. Especially, after finding the histogram peak, a peak shift compensation is often required to reduce the non-linearities, in which the system complexity rises accordingly. By trying to solve the problems directly, a question emerges that is: is there a direct relationship between the bin counts and the actual depth in one histogram? We can associate the bin counts with inputs to a non-linear unknown function, and the depth with its output. Hence, we propose to answer this question with the use of neural network estimation.

In this section, we propose an artificial neural network (ANN) algorithm, which directly calculates the depth hidden in the non-linearities of the system. It can automatically learn to robustly estimate depth. Common problems including duplicate datasets

and over-fitting are evaluated and discussed [6, 7]. In addition, as the algorithm is implemented in an FPGA (see chapter 6). It is also essential to minimize the resource utilization of the ANN.

5.3.1. PROPOSED ANN ARCHITECTURE

The goal of this ANN model is to estimate the depth. Thus, the network is defined as a signal estimation neural network. Based on the general concept from the previous section, we first define the input and the output of the network, in which the input are the counts of the 256 bins of one histogram, and the output is the target depth. Therefore, the depth estimation process can be summarized as a non-linear transformation from $\mathbb{R}^{256}$ to $\mathbb{R}^1$, and it is given by

$$\mathbb{R}^{256} \Longrightarrow \mathbb{R}^1$$
$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{256} \end{pmatrix} \Longrightarrow \left(\hat{D}\right), \tag{5.4}$$

where $y_1, y_2, ..., y_{256}$ represents the counts at every bin of the histogram, and $\hat{D}$ is the estimated depth value.

Usually, when the relation between inputs and outputs is linear, there is no need to use hidden layer with non-linear transfer functions; for mapping the inputs in one finite space to the outputs in another finite space, one hidden layer can be applied to approximate the function [8, 9]. For our application, we assume that one hidden layer is sufficient for the depth estimation task.

Next, a question arises that how many neurons are required in the hidden layer. From previous works, some criteria were provided but none were accurate [9]. However, in practice, we swept the number of hidden neurons and found that 8 neurons for the hidden layer is a sufficient value for our model.

Therefore, the complete architecture of the ANN is proposed and shown in Fig. 5.6. Counts at each of the 256 histogram bins are fed into the architecture model, and a depth value is defined as the output. The architecture contains one hidden layer of 8 neurons and one output layer of 1 neurons. Each neuron has its own weights (W) and biases (B). The hidden neurons utilize non-linear activation function, while the output neuron uses a linear activation function.

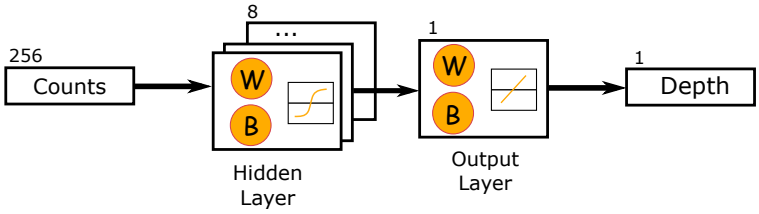


Fig. 5.6: Proposed ANN architecture.

5.4. WORKFLOW

In general, an ANN algorithm workflow includes training and testing stages. The training stage determines the weights and biases values of each neuron of the algorithm, and testing stage tests the effectiveness of the trained network. In this section, we present the workflow of the algorithm, focusing mainly on the training stage.

Essentially, the training stage contains five steps, including

1. initialization of the network,
2. perform forward propagation of the training set,
3. compare the network output with the expected output,
4. perform backward propagation to update weight and bias.
5. repeat this process iteratively, from the second point, until a predefined training condition.

The steps mentioned above can be expressed in global formulas. In the following, they are illustrated by taking a simplest ANN algorithm containing one hidden layer as an example. And it is assumed that its output layer has one output neuron.

Starting from initialization, in which every weight value is randomized between -1 to 1, bias values are set to 0. And the input dataset is normalized between -1 to 1. They are given by

$$\begin{aligned} \mathbf{W_1}(i,j) &= \text{random}[-1,1] \quad && i \in \{1,2,...,I\}, j \in \{1,2,...,J\} \\ \mathbf{W_2}(k) &= \text{random}[-1,1] && k \in \{1,2,...,K\} \\ \mathbf{B_1}(l) &= 0 && l \in \{1,2,...,L\}, \\ \mathbf{B_2} &= 0 && - \\ \mathbf{y_n'} &= \text{norm}(\mathbf{y}) && - \end{aligned} \tag{5.5}$$

where $\mathbf{W_1}$, $\mathbf{B_1}$ indicate the weight and bias for the hidden layer, $\mathbf{W_2}$, $\mathbf{B_2}$ indicate the weight and bias for the output layer, and $\mathbf{y_n'}$ is the normalized transpose vector of $\mathbf{y}$, in which the row vector $\mathbf{y} = [y_1, y_2, ...y_n]$.

Next, the forward propagation step starts. It is provided by

$$\hat{D} = f_2(\mathbf{W_2} \cdot (f_1(\mathbf{W_1} \cdot \mathbf{y_n'} + \mathbf{B_1})) + \mathbf{B_2}), \tag{5.6}$$

where $\hat{D}$ is the predicted depth, f_1 represents the activation function for the hidden neurons, and f_2 is the linear transfer function for the output neurons.

The training output is then compared with the target output, namely 'loss'. For example, the MSE operation is used to compute the loss (L), given by

$$L = \frac{1}{2N} \sum_{i=1}^{M} (\hat{D}_i - T_i)^2, \tag{5.7}$$

where T_i is the target output, N is the total number of observations and M is the amount of training output.

The backward propagation is followed to update the weight and bias, focusing on the derivative of the loss with respect to the weights and bias. Typically, a gradient descent ($\Delta w_{i,j}$, $\Delta B_{i,j}$) works together with the learning rate (α), reflecting the impact of the L on the weight and bias. It is provided by

$$\begin{aligned} \Delta W_{i,j} &= -\alpha \frac{\delta L}{\delta W_{i,j}} \\ W'_{i,j} &= W_{i,j} + \Delta W_{i,j} \\ \Delta B_{i,j} &= -\alpha \frac{\delta L}{\delta B_{i,j}} \\ B'_{i,j} &= B_{i,j} + \Delta B_{i,j} \end{aligned} \tag{5.8}$$

Finally, the global training algorithm will stop, until the algorithm performance meets the performance conditions set by the designer. After the ANN is trained, designers can test the performance as well as extracting the key features to evaluate if the training was successful or not.

The testing stage utilizes a set of testing patterns ($\mathbf{V}$) to test the effectiveness of the network. It is provided that

$$\hat{\mathbf{T}} = \text{test}(\mathbf{V}). \tag{5.9}$$

In the following subsections, the training and testing stages are described in detail, which are implemented in MATLAB.

5.4.1. ANN TRAINING

For an ANN training process,the first step is to have sufficient data to build the training dataset. And, the training set should be composed of representative samples of the application. In our design, the dataset is generated from the 1D-TOF LIDAR simulation model.

As we already define the column vector $[y_1, y_2, ...y_{256}]^{\mathrm{T}}$ as the input of the neural network, sufficient vectors that cover all available input patterns are required. Hence, we utilize MATLAB to run the optical and statistical simulation by sweeping the parameters linearly to obtain representative datasheet. At this moment, we assume that the feature for the internal system is already fixed, and, sweep only the non-adjustable parameters (see table 4.2). Background noise, target reflectivity and target distance are three external parameters that have impact on the output histogram. Based on the value from the specs (see table 3.2), we prepare the data accordingly. The swept parameters and their range can be found in table 5.3, in which Step-Train represents the steps when creating the train set, and Step-Test for creating the test set. It is important to mention that at every parameter condition, the simulation is repeated for 5 times, in order to cover the effects from statistical factors.

In this way, we obtain over 300,000 samples to build the training set (D_TR). The distance value for every sample is recorded accordingly as the target depth (T_TR).

We also create a dataset for the testing in the same way. By applying another distance sweeping step, which is non-integer multiples of to the step when creating the training

Parameter	Start	End	Step-Train	Step-Test
E_e	0	0.4 W/m^{2}nm	0.05 W/m^{2}nm	0.05 W/m^{2}nm
ρ	8%	60%	1%	1%
d_T	0.1 m	0.6 m	0.0025 m	0.01 m

Table 5.3: Range of swept parameters in ANN dataset. At every parameter combination, the simulation is repeated for 5 times. The adjustable parameters are: TCSPC = 30,000, TDC LSB = 25 ps, P_s = 7.36 mW.

set (see table 5.3). The testing set (D_TE) contains over 90,000 samples, whose distances are recorded in matrix T_TE.

Next, the proposed ANN is trained by utilizing MATLAB. Its *Neural Network Toolbox* calls the *feedforwardnet* function, in which way an ANN is created [10]. A general setting of the object properties is shown in table 5.4, in which the training algorithm workflow is implemented. Other parameters are kept as default.

Property	Function/Value	Description
net.initFcn	initlay	Initialization function
net.inputs{1}.processFcns	mapminmax	Input normalization function
net.outputs{2}.processFcns	mapminmax	Output de-normalization function
net.layers{1}.transferFcn	tansig	Hidden layer activation function
net.layers{2}.transferFcn	purelin	Output layer activation function
net.performFcn	mse	Loss function
net.trainFcn	trainlm	Network training function
net.trainParam.epochs	1000	Max. training epoch E_{p0}
net.trainParam.min_grad	1e-7	Min. gradient Δ_0
net.trainParam.max_fail	6	Max. validation check n_0
net.trainParam.time	Inf	Max. running time T_0
net.trainParam.goal	0	Min. Loss L_0

Table 5.4: MATLAB *feedforwardnet* property setting.

The function is then called, creating and training the ANN as following codes:

Listing 5.4: ANN training MATLAB code.

```
net = feedforwardnet(8,'trainlm');
[net,tr] = train(net,D_TR,T_TR);
```

where '8' is the number of neurons in the hidden layer, and 'trainlm' is taken as the backward propagation algorithm to calculate the Jacobian matrix that leads to $\Delta w_{i,j}$, $\Delta B_{i,j}$ in equation (5.8).

Essentially, the implemented training algorithm will stop, until one of the following condition happens:

1. training epoch is greater than a set value E_{p0};
2. training time is greater than a set value T_0;

3. training loss L is less than a target value L_0;
4. updated gradient $\Delta w_{i,j}$, $\Delta B_{i,j}$ is less than a set value Δw_0, ΔB_0;
5. the loss L of an individual dataset keeps steady for n_0 times (see table 5.4)

At the stop of the training process, it must be ensured that no overfitting has occurred. The MATLAB *Neural Network Toolbox* allows us to observe the performance of both the validation and training sets. We can determine whether overfitting occurs by observing whether their corresponding curves diverge when training process is stopped. However, in general, overfitting does not occur during this process. Because MATLAB has a default early stopping function, training is interrupted when overfitting starts to occur [11].

5.4.2. ANN TESTING

To evaluate the effectiveness of the implemented training algorithm, we use testing set (D_TE), generated by the parameter sweep in table 5.3, to test the network. The testing process is coded as:

Listing 5.5: ANN testing MATLAB code.

```
1 DEPTH = net(D_TE);
```

where DEPTH is the estimated depth matrix.

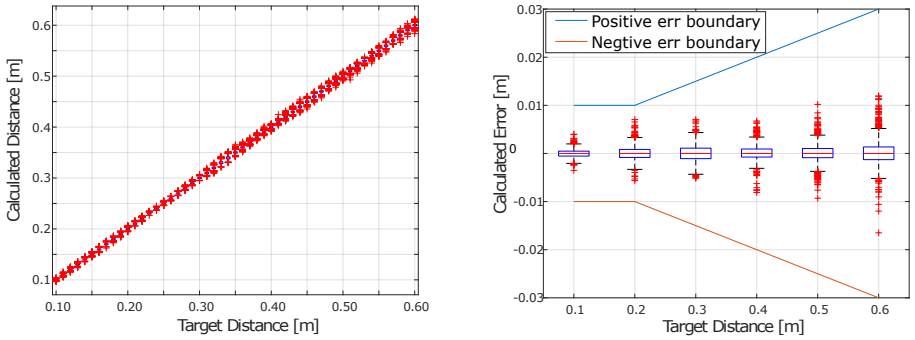
We want to know, the performance of the ANN to input histograms that were not used during training. Considering the way we generate the testing dataset, every possible pattern at one distance point is created individually. We call all the patterns at one distance point as a 'subset', in which a 'subset' contains all possible patterns at that distance point. To plot out the simulation results, a box-plot is made. At every distance point, the box contains the distribution of estimated depth values for the whole subset. The entire box-plot is shown in Fig. 5.7(a).

Distance checkpoint [m]	Estimated Depth [m]
0.1	0.1000±0.0016
0.2	0.1999±0.0028
0.3	0.3000±0.0033
0.4	0.4000±0.0029
0.5	0.4998±0.0035
0.6	0.6001±0.0049

Table 5.5: Estimated depth summary using $\mu \pm 2\sigma$.

The most important observation from the simulation result is that, the ANN algorithm never fails in estimating the depth. To make the plot readable, we select six groups in the plot with linear step, and calculate the estimated depth as well as the relative error of the data in those groups.

The estimated depth is also expressed in the form of $\mu \pm 2\sigma$, as introduced in equation (5.3). The result can be found in table 5.5.



(a) (b)

Fig. 5.7: ANN testing simulation result. (a) Estimated depth vs. target depth; (b) Estimated error vs. target error.

The improved box-plot is made as follows. First, every group subtract the group median, to normalize the center of the box to 0. Next, we create the error boundary according to the specs (see table 3.2), and compare the normalized box-plot with respect to the error boundary (see Fig. 5.7(b)). An obvious result is that the error between the estimated depth and the target depth is within the allowable range as far as the specification is concerned.

5.4.3. PERFORMANCE EVALUATION

It is essential to mention first that the training dataset plays an crucial role in the algorithm. Indeed, a well-functional neural network requires sufficient training before putting into testing. One needs to train the network to let it recognize possible data patterns. Hence, having a training dataset that covers all possible cases is the guarantee of the success of the ANN.

In our application, once the proposed ANN is well trained, one can obtain the estimated depth $\hat{D}_e$ by just feeding the count-vector under test (**CvUT**) to the network, and apply forward propagation, once. It can be given that

$$\hat{D}_e = f_2(\mathbf{W_2} \cdot (f_1(\mathbf{W_1} \cdot \mathbf{CvUT} + \mathbf{B_1})) + \mathbf{B_2}), \tag{5.10}$$

where $\mathbf{W_1}$, $\mathbf{W_2}$, $\mathbf{B_1}$ and $\mathbf{B_2}$ are the trained weight and bias from the neural network. A hyperbolic tangent sigmoid function f_1, and a linear function f_2 are the layer transfer functions.

As we can also notice, except for a non-linear transfer function f_1, all the forward propagation process only requires additions and multiplications. This makes it possible to implement a well-trained algorithm into the field-programmable gate array (FPGA).

We also want to compare the performance of the ANN to the other two algorithms proposed in this chapter. First of all, the subtraction algorithm is not possible to work alone to identify the peak, which only works for filtering the noise. For the ANN and the

PD algorithm, we can compare their performance from Fig. 5.7(b) and Fig. 5.4. One crucial point is that the PD algorithm may fail in 0.4 to 0.6 m distance range, but the ANN does not. Considering the robustness of the algorithm, the ANN is better than the PD algorithm.

5.5. DISCUSSION

In this chapter, three algorithms including PD, subtraction and ANN are introduced and evaluated. The final goal of the algorithm, is to identify the depth from the timestamping histogram. Based on this goal, we compare the performance of the three algorithms.

The PD algorithm is intuitive, but it does not work well near maximum distance of 0.6 m. And it requires an extra compensation to offset the effects originated from the system non-linearity. Also, the PD algorithm requires one more step to perform the correction to compensate the peak shift effect, while the ANN automatically compensates any non-linearity.

The Subtraction algorithm acts only for filtering the noise, which also needs to work together with the PD algorithm to accomplish the final goal. The ANN is robust and accurate in the detection range, and the trained ANN can give valid depth without the need of noise filtering. Hence, we conclude that the ANN is the most effective algorithm.

BIBLIOGRAPHY

[1] MATLAB interp1. *1-D data interpolation, version 9.12.0.1884302 (R2022a)*. (Last accessed: 31/05/2022). The MathWorks Inc., 2022. URL: https://nl.mathworks.com/help/matlab/ref/interp1.html?searchHighlight=interp1&s_tid=srchtitle_interp1_1.

[2] MATLAB findpeaks. *Find local maxima, version 9.12.0.1884302 (R2022a)*. (Last accessed: 31/05/2022). The MathWorks Inc., 2022. URL: https://nl.mathworks.com/help/signal/ref/findpeaks.html?searchHighlight=findpeaks&s_tid=srchtitle_findpeaks_1.

[3] Augusto Ronchini Ximenes. "Modular time-of-flight image sensor for light detection and ranging: A digital approach to LIDAR". English. PhD thesis. Delft University of Technology, 2019. DOI: 10.4233/uuid:c434368a-9a67-45de-a66f-f5dc30430e03.

[4] Preethi Padmanabhan, Chao Zhang, and Edoardo Charbon. "Modeling and Analysis of a Direct Time-of-Flight Sensor Architecture for LiDAR Applications". In: *Sensors* 19.24 (2019). ISSN: 1424-8220. DOI: 10.3390/s19245464. URL: https://www.mdpi.com/1424-8220/19/24/5464.

[5] Gianluca Agresti and Pietro Zanuttigh. "Combination of Spatially-Modulated ToF and Structured Light for MPI-Free Depth Estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. Sept. 2018.

[6] Erzhou Zhu et al. "DTOF-ANN: An Artificial Neural Network phishing detection model based on Decision Tree and Optimal Features". In: *Applied Soft Computing* 95 (2020), p. 106505. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.106505. URL: https://www.sciencedirect.com/science/article/pii/S1568494620304440.

[7] El-Sayed M El-Alfy. "Detection of Phishing Websites Based on Probabilistic Neural Networks and K-Medoids Clustering". In: *The Computer Journal* 60.12 (Apr. 2017), pp. 1745–1759. ISSN: 0010-4620. DOI: 10.1093/comjnl/bxx035. eprint: https://academic.oup.com/comjnl/article-pdf/60/12/1745/22297461/bxx035.pdf. URL: https://doi.org/10.1093/comjnl/bxx035.

[8] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[9] Saurabh Karsoliya. "Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture". In: *International Journal of Engineering Trends and Technology* 3.6 (2012), pp. 714–717.

[10] MATLAB feedforwardnet. *Generate feedforward neural network, version 9.12.0.1884302 (R2022a)*. (Last accessed: 31/05/2022). The MathWorks Inc., 2022. URL: https://nl.mathworks.com/help/deeplearning/ref/feedforwardnet.html?s_tid=doc_ta.

[11] MATLAB. *Improve Shallow Neural Network Generalization and Avoid Overfitting, version 9.12.0.1884302 (R2022a)*. (Last accessed: 31/05/2022). The MathWorks Inc., 2022. URL: https://nl.mathworks.com/help/deeplearning/ug/improve-neural-network-generalization-and-avoid-overfitting.html.

6

SYSTEM IMPLEMENTATION

Mingzhe CHEN

In this chapter, the proposed 1D-TOF system is implemented on an FPGA. The system is designed in a modular structure. In the following sections, the top-level block diagram is first introduced, which is then divided into sub-modules that are described in detail. The characterization results are provided later, which were measured utilizing the integrated-logic-analyzer (ILA) available in the FPGA.

6.1. INTRODUCTION

In the previous chapters, the 1D-TOF LIDAR system modeling and the TOF processing algorithms were introduced. In this chapter, the proposed system and the trained ANN algorithm are decomposed into sub-modules and implemented on an FPGA.

The proposed 1D-TOF system is implemented on a Xilinx Kintex-7 KC705 evaluation kit that is connected to a printed circuit board (PCB) from Silicon Integrated (SI) B.V. [1], which contains a SPAD sensor. The system sub-modules control the sensor and perform the depth estimation task to give the estimated distance.

The trained ANN algorithm is also implemented in the FPGA for real time processing. It is worth mentioning that the depth estimation algorithm sub-module serves as a stand-alone and replaceable system component. This means that one can switch to different algorithms to obtain the depth, just by connecting the ports to a new algorithm sub-module.

[1] The PCB board contains a vertical-cavity surface-emitting laser (VCSEL) and Single-photon avalanche diodes (SPADs). Its pins are designed specially for high-pin-count (HPC) connectors on the KC705 evaluation kit. Silicon Integrated Co., Ltd: https://www.si-in.com

6.2. BLOCK DIAGRAM

In chapter 2, a prototype system architecture was introduced. Fig. 6.1 shows a detailed part of the architecture that is implemented on the FPGA. The SPAD pixel and VCSEL driver are implemented in the SI prototype PCB. A simplified block diagram is made, in order to show the connections between the FPGA and the SI prototype board (see Fig. 6.2).

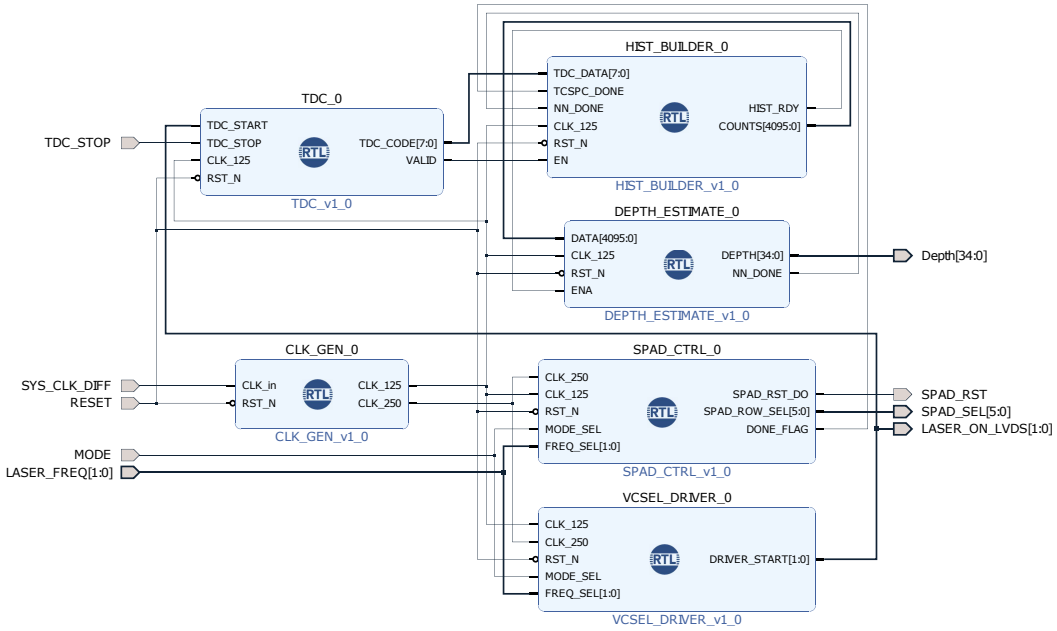


Fig. 6.1: 1D-TOF system architecture. This diagram was generated with the block design tool of Xilinx Vivado.

The modular architecture contains five input ports, four output ports and six sub-modules at the top level. The modules can be divided into three types, including clock distribution, system control and data processing.

One of the main signal input ports is called TDC_STOP. It is the response signal from the SPAD circuity, which is utilized by the time-to-digital (TDC) module to calculate the timestamp of a detection. In addition, a 200 MHz system clock and a reset port are included in the top level ports. They serve as global clock source and reset function. Moreover, there are other two ports called MODE and LASER_FREQ. Port MODE serves as the SPAD reset selection. And, port LASER_FREQ selects the laser repetitive frequency, with 5, 10, 15, 20 MHz.

The output ports have three control execution ports and one final depth output. The control execution ports control the laser and the SPADs. It is important to mention that, the output port called SPAD_SEL can enable the selected SPAD [2]. The SPAD_RST is used

[2] On the PCB board, there is a SPAD array of 40 × 30 SPADs. The SPAD_SEL port will specify a row, and we can choose 1 SPAD out of 30 to record its response.

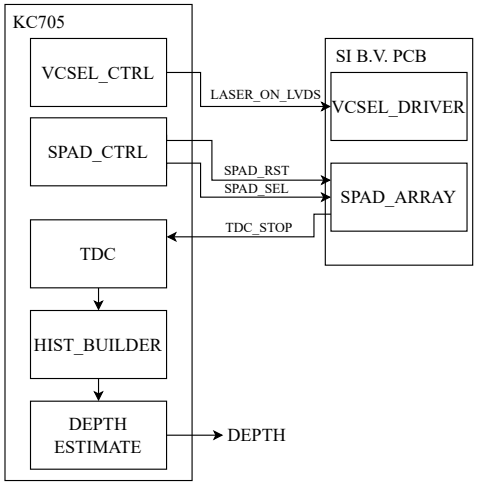


Fig. 6.2: A simplified block diagram: connections between KC705 FPGA and SI B.V. PCB. The clock module on FPGA is not presented.

for resetting the SPADs, and the LASER_ON_LVDS is connected to the input of the VCSEL driver. The final depth output is the final result of the entire on-board system, which is the estimated depth coded in binary.

The six modules of the top level architecture contain the clock generation, PCB control and data processing functions. The clock generation module is implemented with the internal phase lock loop (PLL) of the FPGA, with 200 MHz clock oscillator connected at the input [3]. A 125 MHz and a 250 MHz output is generated by the PLL, corresponding to the nets CLK_125 and net CLK_250 (see Fig. 6.1). The other five modules are explained in the following sections.

6.3. VERTICAL-CAVITY SURFACE-EMITTING LASER DRIVER

The vertical-cavity surface-emitting laser (VCSEL) is included on the PCB board from SI B.V.. The control process of the VCSEL driver is introduced in the following paragraphs.

To drive the VCSEL, a 2 ns low-voltage differential signaling (LVDS) pulse width is required [4]. In the VCSEL driver, a counter works under 125 MHz clock, to approximate the period under the specified laser repetition frequency. A period conversion can be found in table 6.1. The effect of the repetitive frequency on the timing of the LASER_ON signal is shown in Fig 6.3. It is important to mention that the VCSEL repetitive frequency is different from that in the specs in table 3.2. And, this group of 5 MHz to 20 MHz repetitive frequency is only used for sub-module validation.

Next, the module generates a half-cycle pulse under 250 MHz clock domain every required cycle (the second column in table 6.1). The single-ended pulse further requires

[3] On KC705, the board has a 2.5V LVDS differential 200 MHz oscillator, which is Si Time SiT9102AI-243N25E200.00000 (200 MHz)

[4] The value 2 ns is the full-width-half-maximum (FWHM) of the pulse.

VCSEL repetitive frequency	Cycles required under 125 MHz clock	Description
5 MHz	25	25 = 125/5
10 MHz	13	13 ≈ 125/10
15 MHz	8	8 ≈ 125/15
20 MHz	6	6 ≈ 125/20

Table 6.1: Period conversion table.

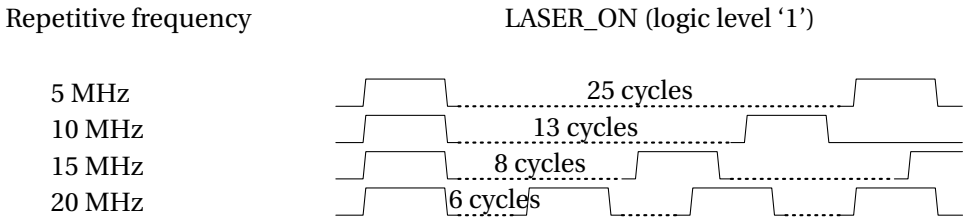


Fig. 6.3: VCSEL repetitive frequency selection timing logic. Every logic level '1' represents the moment that the single-ended LASER_ON signal is activited. And, 'cycles' represents the period cycle under 125 MHz clock.

a differential output buffer module (OBUFDS) to convert it into LVDS.

6.4. SINGLE-PHOTON AVALANCHE DIODE CONTROL

The SPAD control module has two major functions, including reset the SPAD before each detection cycle, and select the desired SPAD from the SPAD array. It works simultaneously with the VCSEL driver.

For SPAD reset, the logic table of the reset choice is given in table 6.2. Once MODE is set as 0, the SPAD will be reset 8 ns before the VCSEL driving pulse is emitted. Otherwise the SPADs will be passive quenched once it is triggered by a photon [1].

MODE	Reset SPAD before VCSEL	Description
0	On	Default
1	Off	-

Table 6.2: SPAD mode logic table.

As for the selection of the SPAD out of the SPAD array, a block diagram of the SPAD array is shown in Fig. 6.4. It is worth mentioning that, here we utilize a different SPAD array comparing to the SPAD array mentioned in the specs from table 3.2, due to availability issues. Also, here we present the connection showing one activated SPAD, which will futher be connected to one single TDC. To be specific, the SPAD array itself has 40 rows, and there are 30 individual SPADs in each row. The SPAD_SEL can specify a row entirely to the output SPAD_out0 to SPAD_out29. As we only need 1 SPAD in this case, a default selection is made, in which the first row is selected, and the middlemost SPAD output is connected to the TDC.

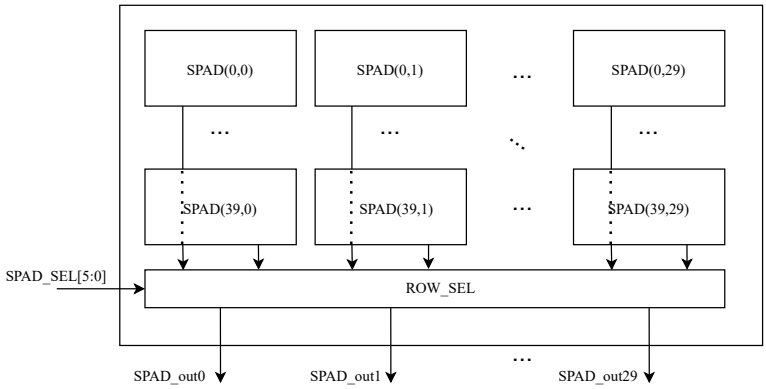


Fig. 6.4: SPAD array block diagram.

6.4.1. HARDWARE VALIDATION

To check if the control logic can work independently before the data processing sub-modules are implemented, we perform hardware validation, which includes a validation with an ILA core and raw signal detection with oscilloscope (see Fig. 6.5). It is important to note that the dashed-line modules are not included in the validation, and the TDC_STOP is connected to a float pin on the FPGA.

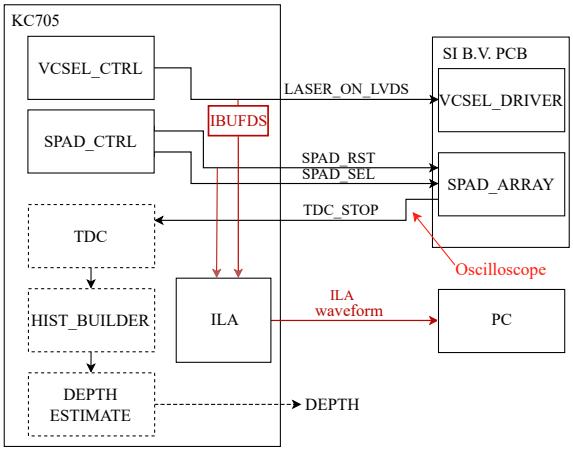


Fig. 6.5: Block diagram: SPAD control and VCSEL driver with ILA setup.

VALIDATING OPERATION

An ILA core is added in the design, which can probe the net and record the samples. It is worth mentioning that the ILA validates the actual FPGA firmware running in the FPGA, by probing signals under testing, and it is not a simulation study. In our implementation setup, the LASER_ON_LVDS and SPAD_RST signal are probed by the ILA core. The

LASER_ON_LVDS signal is converted into the single-ended signal for easy observation. The ILA core can send the recorded waveform to the personal computer (PC). By observing the results recorded by the ILA core, we can compare the performance to what we expect.

VALIDATION RESULTS

In Fig. 6.6, the waveforms for driving the selected SPAD is shown, in which the VCSEL_ON signal is the VCSEL trigger and SPAD_RST_DO is the SPAD reset signal. In this waveform, the SPAD is set as always reset itself before the VCSEL is on.

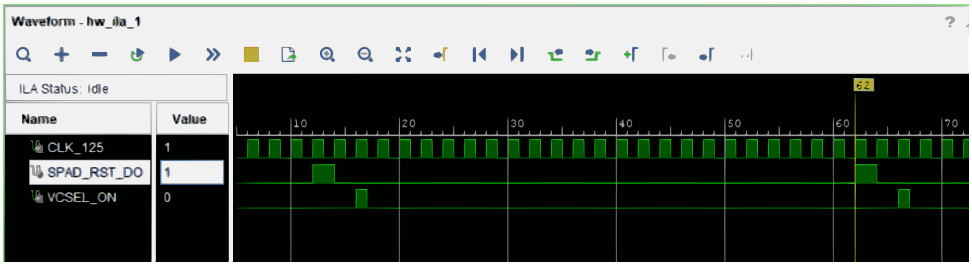


Fig. 6.6: ILA results screenshot for VCSEL & SPAD control.

We also take advantage of an oscilloscope to measure the SPAD outputs (TDC_STOP). The validation result is shown in Fig. 6.7. We can observe that the SPAD is triggered, and its outputs are captured on the screen. In addition, the oscilloscope also accumulates a histogram, which can be observed in this plot.

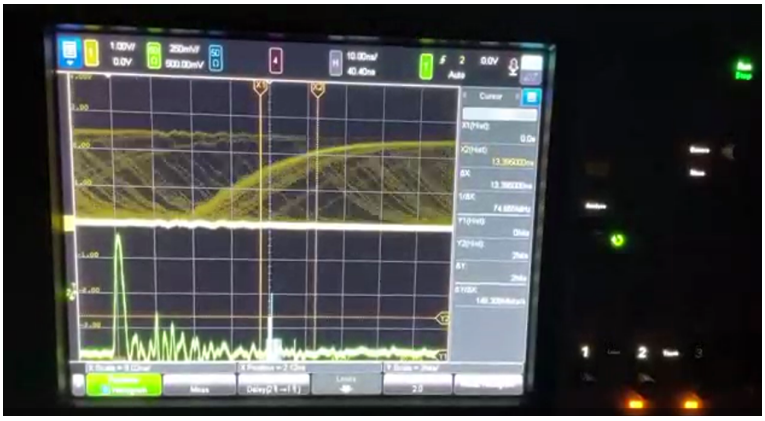


Fig. 6.7: Oscilloscope measurement at SPAD output.

6.5. TIME-TO-DIGITAL CONVERTER

The timestamp information of the SPAD output is acquired by the time-to-digital converter (TDC). In this project, one single TDC is implemented on FPGA, connecting the

single SPAD from the SPAD array to the stop signal, while the start signal is the single ended LASER_ON signal.

6.5.1. BLOCK DIAGRAM

In general, FPGAs have predefined structures, such as slices, PLLs, block RAMs etc. It is possible to implement the core component of a TDC, which is a delay chain, utilizing the carry chain of the FPGA's slices. In our design, we implement the TDC based on the carry logic of the KC705 evaluation kit.

The architecture of the TDC is given in Fig. 6.8, which contains four major sub-modules, including a coarse counter, a fine counter, a latch group and a decoder. The inputs of the main TDC itself are the start and stop signals, and the outputs contain the digital code and a valid signal. Additionally, the TDC have extra control signals, such as reset, clk, etc.

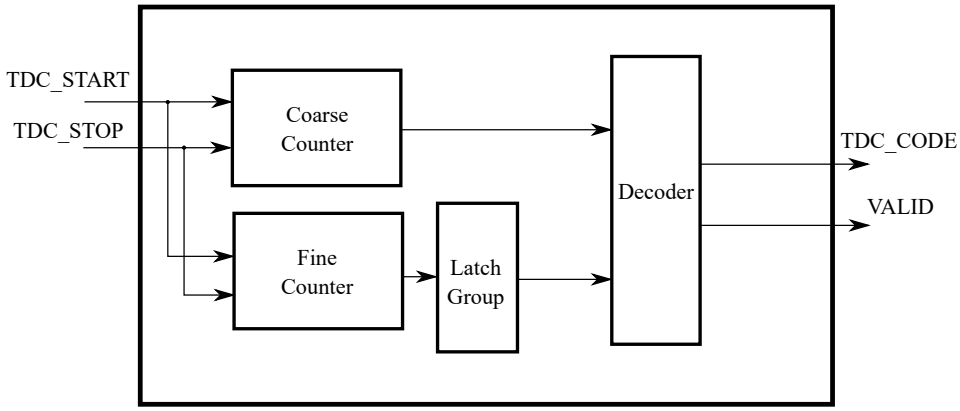


Fig. 6.8: Block diagram of TDC module.

A timing diagram can be found in Fig. 6.9. It is worth mentioning that, in our implementation, the start signal of the TDC is the driver signal of the VCSEL. Hence, the rising edge of the start signal is the same as the main 125 MHz clock. The stop signal is connected to the SPAD output, which is considered as an asynchronous input. The coarse counter calculates T1, which is the integer multiples of one clock cycle. T1 starts at the rising edge of the start signal, and it ends at the next rising edge of the clock right after the rising edge of the stop signal. The fine counter records the T3, which is the time between the rising edge of the stop signal and its next clock's rising edge. Thus, the time T2 is then calculated as:

$$T2 = T1 - T3. \tag{6.1}$$

To be more specific, the coarse counter is implemented as a period counter of the system clock. And, the fine counter is implemented by the carry chain logic, in which the delay of the carry logic serves as the least-significant-bit (LSB) of the TDC. In this design, it is also essential that the delay chain should cover one clock cycle to avoid sparkle-

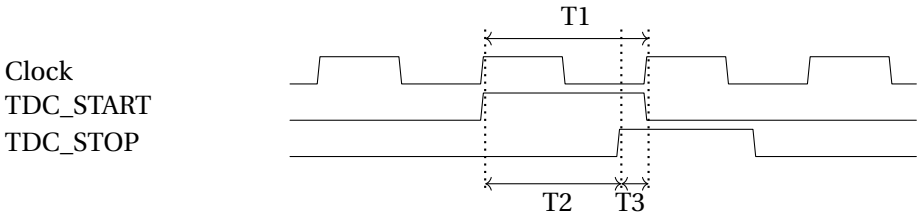


Fig. 6.9: TDC timing logic.

code [5]. The latch group records the values fine counters every time the TDC is triggered by a stop signal. And, the recorded values are then proceeded by a decoder to obtain a corresponding digital code.

6.5.2. CHARACTERIZATION

To characterize the TDC, we perform basic characterization evaluation about the TDC full scale range and its least-significant-bit (LSB).

TDC FULL SCALE RANGE

The TDC full scale range depends on T1, which is the coarse period counter (see Fig. 6.9). In the design, we set the full scale range is 64 ns, which is exactly 8 cycles of the clock period [6].

TDC LSB

The TDC LSB is the cell delay in the fine counter, which is the carry-delay of a carry chain in our design. To measure the LSB of the TDC, we need to further understand the architecture of the TDC on FPGA. One essential step is to fix the location within the FPGA available slices of the carry chain of the fine counter, in order to ensure a same delay between the carry chain cells. Therefore, in the design, we fix the carry chain as a column, as shown in Fig. 6.10.

Next, we input signals in the start and stop pins of the TDC with 2 ns, 4 ns and 8 ns difference. And, measure the value recorded by the carry chain. And the fine counter result T3 is given by

$$T3 = T1 - T2, \tag{6.2}$$

$$T3 = NT_0, \tag{6.3}$$

where T_0 is the unit delay of the cell, known as LSB in our TDC, and N is the number of the carry cells it passed in the simulation. It is important to mention that the system clock period is 8 ns, that the T1 for 2 ns and 4 ns inputs is 8 ns. But an input of T2 = 8 ns has the same rising edge as the system clock, in which it triggers the setup time violation. And, it will cause the coarse counter to count one more clock cycle, until the next rising

[5] It means that an incorrect value is given by the delay chain, in which the digital code does not cover a certain range of the time.

[6] The set of 64 ns full scale range is to cover potential circuitry delay.

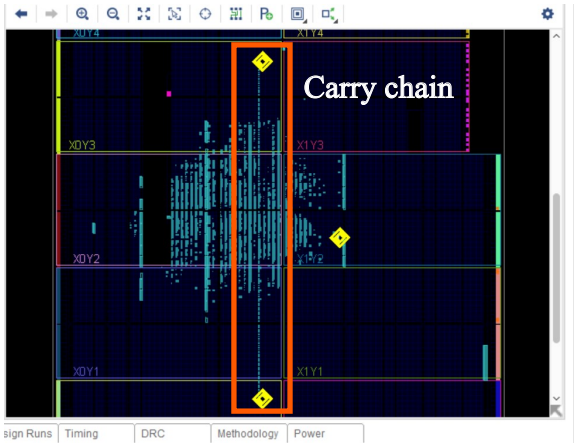


Fig. 6.10: Implemented design device mapping of the TDC. The yellow mark shows the first and the last cell of the carry chain.

edge of the system clock. Thus, the T1 for T2 = 8 ns input is 16 ns. The result of this experiment is recorded in table 6.3.

Input (T2) [ns]	T1 [ns]	T3 = T1-T2 [ns]	Number of carry cells it passed (N)
2	8	6	328
4	8	4	154
8	16	8	492

Table 6.3: Measured number of delay cells using 2 ns, 4 ns and 8 ns inputs.

To remove the circuit delay or the TDC offset from the LSB estimation, we subtract each two of them to measure the delay for a carry cell. And, the LSB is 11.5 ps.

6.6. HISTOGRAM BUILDER

The histogram builder (HB) module creates the timestamp histogram that is used as the input data of the depth estimation algorithm. In the digital approach of this report, its function is achieved by using random access memory (RAM) on the FPGA as core components, which are available in the FPGA.

To group the numeric timestamps, an essential step is to define the range of the bins. Based on the feature of the TDC, in theory, the bin size should be equal to the TDC LSB, which is 25 ps. And there are in total 256 bins, which corresponds to the TDC full scale range. Therefore, whenever a TDC binary code is fed to this HB module, its corresponding bin will perform an accumulation of 'plus one'. The behavior can be depicted as follow, in simplified Verilog:

Listing 6.1: Key ports and behavior description of HB sub-module in Verilog.

```
...
input [7:0] TDC_CODE,
output [15:0] Histogram[0:255],
...
Histogram[TDC_CODE] = Histogram[TDC_CODE] + 1;
```

In the register-transfer level (RTL), the above mentioned behavior is then achieved by utilizing the distributed RAM[7]. The depth of the RAM is 256, so that the TDC digital code can be used to define the address of the RAM. Once a TDC code is available, the module will convert the code into an address. Next, it extracts the stored value from the corresponding RAM address, performs a 'plus one' operation, and stores the updated value back to the same address in the RAM.

6.6.1. HARDWARE VALIDATION

To validate HB module independently, we preload TDC codes in the FPGA memory. And, we compared the results from this module and the histogram generated by MATLAB, to examine the effectiveness of this module. A block diagram can be found in Fig. 6.11, in which the modules with solid lines are implemented for validation.

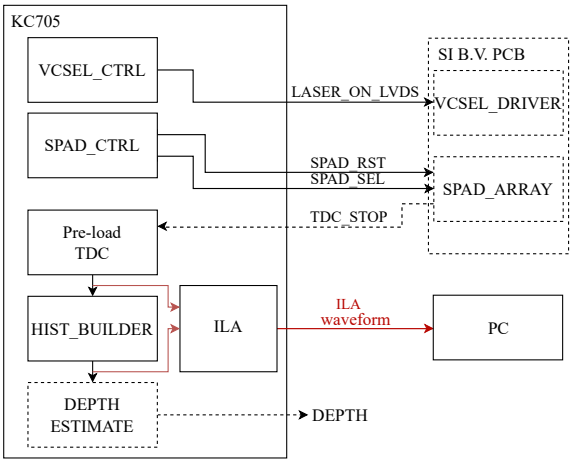


Fig. 6.11: Block diagram: Pre-loaded TDC and histogram builder with ILA setup.

VALIDATING OPERATION (MATLAB)

The way a pre-loaded TDC codes work is the same as a true TDC, but its digital code is already set to be fixed. We generated two sets of TDC codes in advance, which should produce the same histogram. One set is created within MATLAB, and another set is stored in the FPGA configuration file, which effectively acts as a read-only memory (ROM). The MATLAB histogram is elaborated by calling function *hist*. It is given by

[7] The synthesis tool automatically synthesizes the design into distributed RAM using LUT slices on the FPGA.

Listing 6.2: Histogram generation with specified range in MATLAB.

```
BIN_RANGE = 0:TDC_LSB:TDC_FULL_SCALE_RANGE;
TARGET_HIST = hist(TDC_CODE,BIN_RANGE);
```

where BIN_RANGE is defined by the TDC LSB and the full scale range of the TDC. The resulting histogram is recorded as TARGET_HIST.

VALIDATING OPERATION (HARDWARE)

The dataset for RTL is first stored in a ROM. Considering the control logic for VCSEL driver (see section 6.3), the validation logic is defined as, the HB module will read a data from the ROM 10 ns after every time of VCSEL-on. The logic can be found in Fig. 6.12. After all data is read, the final histogram ($\mathbf{H_b}$) is obtained. An ILA screen capture is shown in Fig. 6.13, in which the bin count 0-6 in the final histogram result (COUNT[0-6][15:0]) is shown.

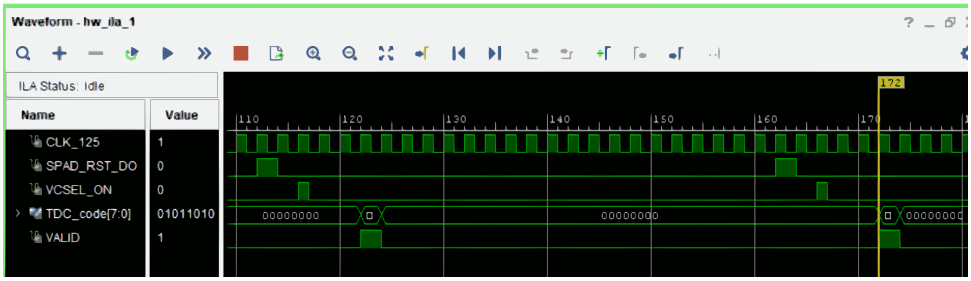


Fig. 6.12: Pre-load TDC logic screenshot from ILA.

VALIDATION RESULTS

In Fig. 6.14, we evaluate the performance of the HB module. By comparing the two histograms probed by ILA and that generated by MATLAB, we observe that they are exactly the same.

6.7. DEPTH ESTIMATOR

The goal of the depth estimator (DE) module is, as its name indicates, to perform the target depth estimation by using the HB output as its input data. From the discussion of the previous chapter, we summarize that a well-trained ANN has the best performance for this estimation task. Therefore, the trained ANN is implemented in the depth estimator sub-module. Moreover, to implement a software algorithm, a general topic is to accommodate the algorithm to the digital signal processor (DSP) algorithm [2]. To have a higher throughput and lower resource usage, an algorithm that uses fixed-point data is required. Hence, it is also essential to evaluate the performance between the fixed-point algorithm and the float-point algorithm.

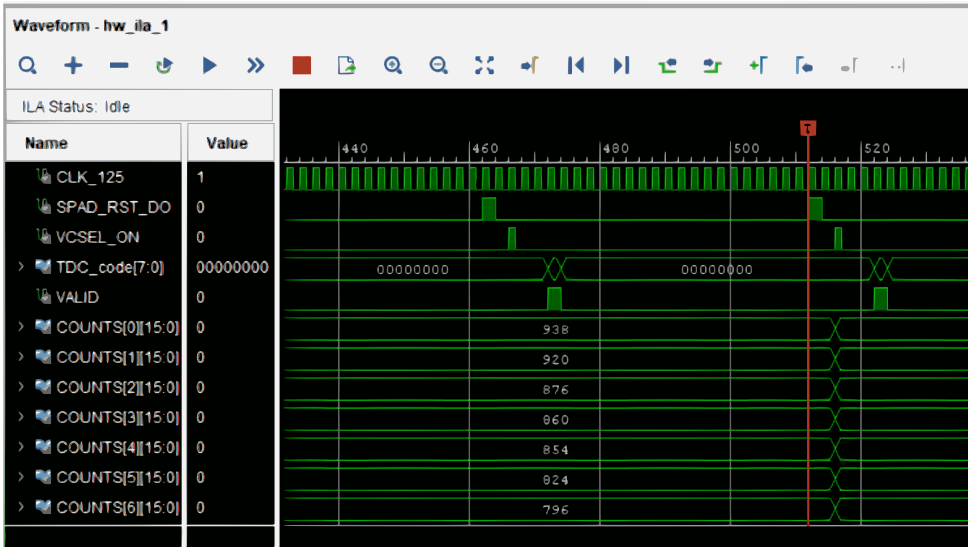


Fig. 6.13: Partial final histogram builder results screenshot from ILA.

6

6.7.1. FIXED-POINT ALGORITHM CONVERSION

To covert the algorithm into fixed-point, it is first essential to identify the float-point data flow in the algorithm. The float-point trained ANN algorithm is shown in algorithm 3, in which it only contains the forward propagation part. The weight (**W**) and bias (**B**) are extracted from the trained ANN in section 5.3.

input : Input column vector $\mathbf{y} = [y_1, y_2, \ldots y_{256}]'$
output : Estimated depth $\hat{D}_e$

$\mathbf{L1\_IN} = \mathbf{W_1} \cdot \mathbf{y} + \mathbf{B_1}$;
$\mathbf{L1\_OUT} = tansig(\mathbf{L1\_IN})$;
L2_IN = $\mathbf{W_2} \cdot \mathbf{L1\_OUT} + \mathbf{B_2}$;
%*The resulting L2_IN is for certain not a matrix.*
L2_OUT = L2_IN;
$\hat{D}_e$ = L2_OUT;

Algorithm 3: Float-point trained ANN forward propagation calculation.

From algorithm 3, three types of data can be found as float-point data, which are **W**, **B** and *tansig*(**L1_IN**) [8]. It is first needed to convert these three types of data into fixed-point.

A common binary fixed-point data format is depicted in Fig. 6.15, in which a word-length including a sign bit, an integer length, a binary point and a fraction length can be found. To represent binary fixed-point data, we need to know at least the wordlength, the sign bit, and the fraction length.

[8] In this ANN algorithm, *tansig* function is the hidden layer activation function (see table 5.4).

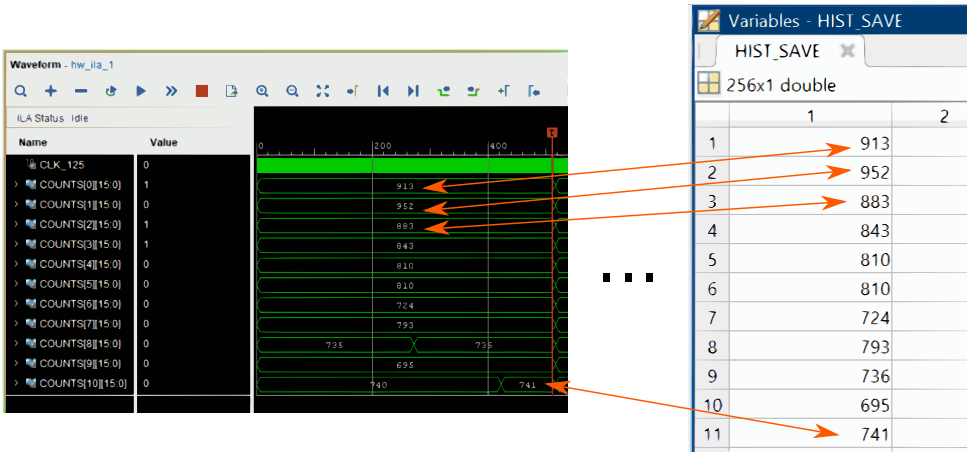


Fig. 6.14: Comparison of ILA resulting histogram and matlab resulting histogram.

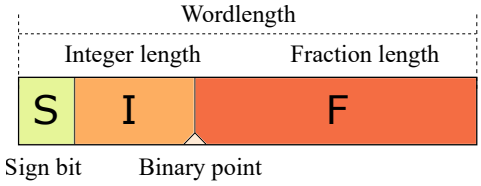


Fig. 6.15: Common binary fixed-point data format.

Therefore, the conversion formula from a float-point data (D_FL) into a binary fixed-point data (B_FI) is provided by

Listing 6.3: Fixed point data conversion MATLAB code.

```
D_FI = round(D_FL.*(2^F)); %F is the fraction length.
B_FI = dec2bin(D_FI,W_0);
```

where D_FI is the fixed-point decimal integer, F is the fraction length, and W_0 is the wordlength. The function *round* can let the value be rounded to its nearest integer, and *dec2bin* converts the decimal format into binary with the specified digits W_0 [3].

Next, we define the rules for data conversion as:

- the converted data integer length (I) must cover the full range of the data,
- sign bit and fraction length can be 0, if the first rule is not violated,
- if implicit leading zeros exists after the binary point and before the binary representation of the stored integer, the fraction length can be greater than the wordlength,
- *tansig* function is replaced by a look-up-table (LUT),

- try to minimize the wordlength, while keeping the fraction length the same [9].

Following the rules above, the data are then converted into fixed-point data. The conversion table is shown in table 6.4. The relative difference for the converted data and the original data is shown in Fig. 6.16, including $\mathbf{W_1}$, $\mathbf{W_2}$, $\mathbf{B_1}$, $\mathbf{B_2}$, and *tansig* output. It is worth mentioning that, there is no need to check the input vector $\mathbf{y} = [y_1, y_2, ...y_{256}]'$, as its values are pure integers so that the fixed-point data does not lose precision during conversion.

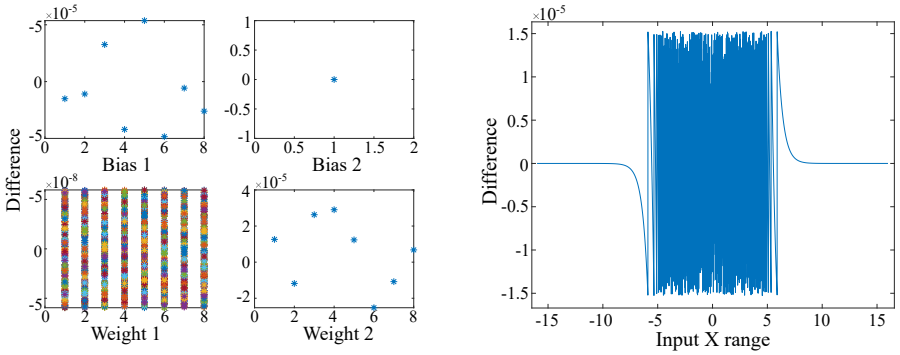


Fig. 6.16: Difference between fixed-point data and float-point data. (a) Weight & bias; (b) *tansig* function.

Data name	Sign bit	Wordlength	Fraction length
y	0	16	0
$\mathbf{W_1}$	1	16	23
$\mathbf{W_2}$	1	16	14
$\mathbf{B_1}$	1	16	13
$\mathbf{B_2}$	1	16	17
tansig LUT output	1	16	15

Table 6.4: Fixed-point conversion.

The converted algorithm is then validated in MATLAB, with the same validation data-set used in section 5.3. It is given that, although the relative error increases compared to the float-point algorithm result, the calculated relative error is still within the allowable range (see Fig. 6.17).

6.7.2. BLOCK DIAGRAM

A block diagram is made in Fig. 6.18 to present the architecture of the DE sub-module, in order to perform the fixed-point algorithm on the FPGA.

[9] An approach to decrease the memory size for further implementation.

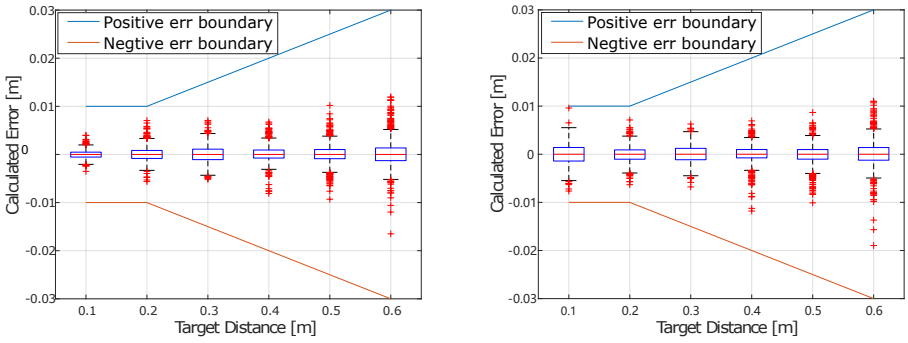


Fig. 6.17: Estimated error comparison. (a) Float point algorithm; (b) fixed-point algorithm.

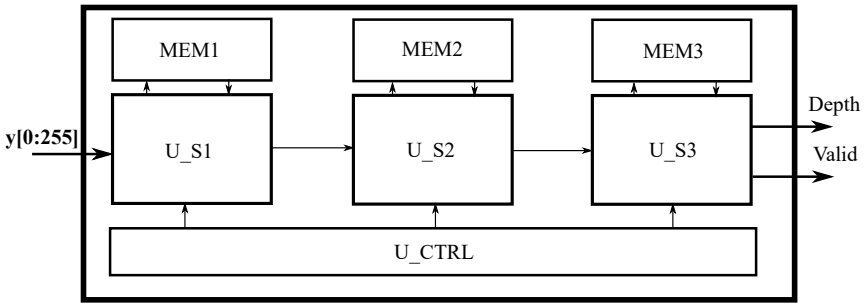


Fig. 6.18: Block diagram of depth estimator module.

RTL abstraction is used to build the sub-modules in Fig. 6.18. It is given that, the input of the DE module is a vector with 256 elements. And the output ports are estimated depth and valid, in which the signal valid is pulled to digital one when the depth estimation is ready.

The sub-module U_S1 operates step 1. It reads MEM1, in which $\mathbf{W_1}$ and $\mathbf{B_1}$ is stored. U_S2 executes the step 2, to perform a *tansig* LUT. MEM2 stores the output values of a *tansig* function, and its addresses are coded as the corresponding inputs of the function. And, U_S3 and MEM3 operates the step 3, calculating with $\mathbf{W_2}$ and $\mathbf{B_2}$. U_CTRL is a control sub-module, which forces U_S1, U_S2 and U_S3 to work pipelined.

6.7.3. TIMING ANALYSIS

From the fixed-point conversion analysis the block diagram overview, we know that it is possible to implement the ANN into the FPGA. The process in algorithm 3 shows that, the calculation contains matrix multiplication and addition, as well as a *tansig* LUT replacement. Therefore, we propose a pipelined timing logic for matrix multiplication and addition. An important constraint is that the algorithm has to be faster than the frame

rate.

In general, the algorithm 3 can be divided into three steps. The overall timing behavior of the module can be described by the timing diagrams in Fig. 6.19, which corresponds to the three sequential steps in algorithm 3. In total, 267 clock cycles are required by the DE module to calculate a single depth point.

It is worth mentioning that, considering the hidden layer weight $\mathbf{W_1}$ is a 8×256 matrix, we assign each row to one DSP block of the FPGA [4], leading to a self-contained multiplication and addition. So, it can also be said that one DSP is engaged in the partial function of a neuron (without the activation function). The nets L1_IN, L1_OUT, Depth correspond to the **L1_IN**, **L1_OUT**, and $\hat{D}_e$ in algorithm 3, respectively.

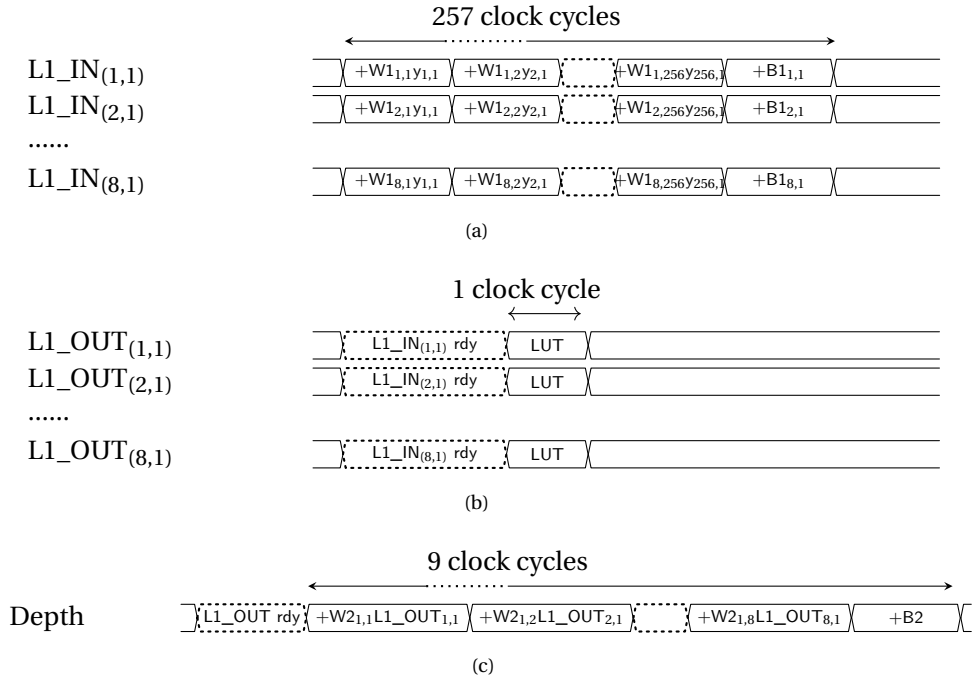


Fig. 6.19: Pipelined timing diagram of the proposed algorithm: (a) Step 1: $\mathbf{L1\_IN} = \mathbf{W_1} \cdot \mathbf{y} + \mathbf{B_1}$; (b) Step 2: $\mathbf{L1\_OUT} = \mathit{tansig}(\mathbf{L1\_IN})$; (c) Step 3: Depth $(\hat{D}_e) = \mathbf{W_2} \cdot \mathbf{L1\_OUT} + \mathbf{B_2}$.

The algorithm is triggered once per frame, in which a frame is considered as finishing every 30,000 cycles of the TCSPC. With the fastest VCSEL repetitive frequency 20 MHz (see table 6.1), the shortest time for one frame is 1.5 ms. Therefore, it is essential to compare the time for the fastest frame rate and the time to achieve the algorithm. It is clear that, under a 125 MHz clock, the algorithm needs 2.136 μs, which is far shorter than the frame time.

6.7.4. HARDWARE VALIDATION

In order to evaluate the performance of the DE module, the hardware validation is performed. A dataset for validation is pre-stored in the FPGA, and the validation dataflow is verified by reading out intermediate FPGA value with the ILA. A block diagram is given in Fig. 6.20, in which the ILA probes are connected to the final output DEPTH, and the internal signals of the DE module.

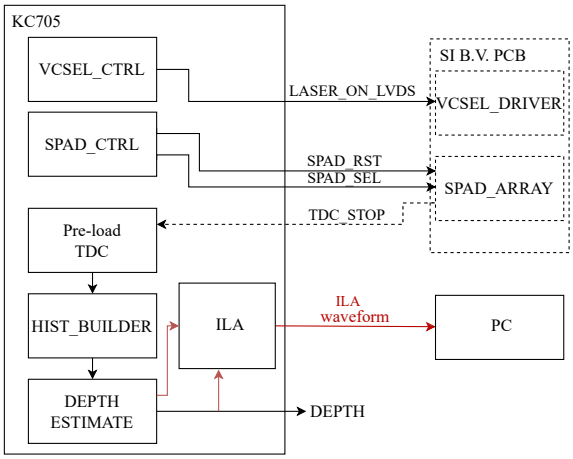


Fig. 6.20: Block diagram: Depth estimator with ILA setup.

VALIDATING OPERATION

The module operates under a clock frequency of 125 MHz. The input histogram comes from the histogram object generated in the section 6.6. The histogram object $\mathbf{H_b}$ is now utilized as the input of DE module. We also put an ILA core to probe the estimated depth, and the outputs at each stage in Fig. 6.18.

VALIDATION RESULTS

The probed results can be found in Fig. 6.21, in which the module outputs the final depth in fixed-point binary code [10] when a valid signal (ALGO_DONE) appears high. The annotations 'Step 1', 'Step 2', and 'Step 3' correspond to the pipelined timing diagram (a), (b), and (c) in Fig. 6.19. Three enable signals (FORWARD1_EN, TANSIG_EN, FORWARD2_EN) ensure the sub-modules to work pipelined.

The probed signal L1_IN and signal Depth in Fig. 6.21 represent the signal with the same name in Fig. 6.19. Here we choose not to probe the result of the *tansig* LUT (L1_OUT) [11].

In table 6.5, we list six groups of depth, which are the actual depth ranging from 0.1 m to 0.6 m, the depth estimated by the model in MATLAB and the depth estimated by

[10] For presentation purposes, this number has been manually converted to hexadecimal in the figure.

[11] The *tansig* LUT need 1 clock cycle in the whole pipelining calculation. And, the DE module runs continuously and periodically under our validation settings. It means that the output of the *tansig* LUT always remains the same. Therefore, the probed output of *tansig* LUT from ILA, is always the same.

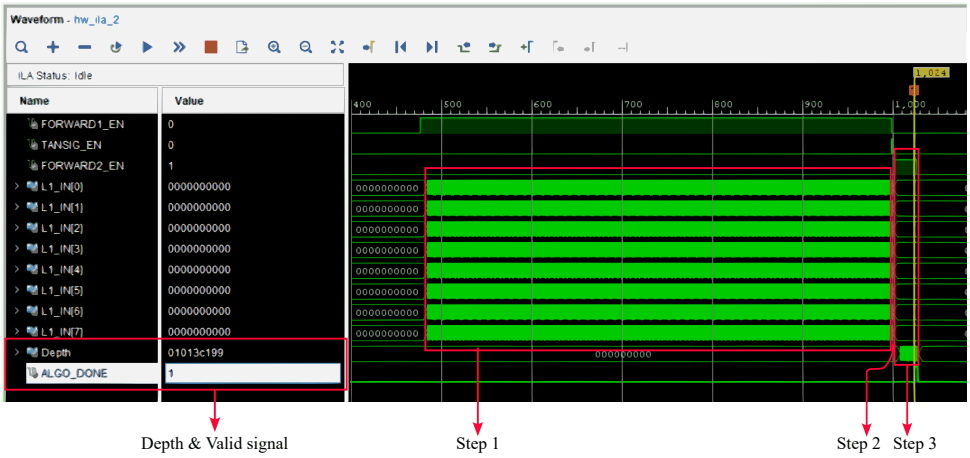


Fig. 6.21: Depth estimator workflow screenshot probed by ILA.

the hardware. At every distance point, we run the validation for 10 different patterns, and the results for estimated depth are shown in $\mu \pm 2\sigma$[12]. We can find that, although the DE module has a higher σ value than MATLAB, the results are still good, which give valid results and acceptable errors.

Distance checkpoint [m]	Estimated depth [m]	
	MATLAB	DE module
0.1	0.1000±0.0016	0.1000±0.0056
0.2	0.1999±0.0028	0.1978±0.0074
0.3	0.3000±0.0033	0.2979±0.0100
0.4	0.4000±0.0029	0.4016±0.0189
0.5	0.4998±0.0035	0.4966±0.0046
0.6	0.6001±0.0049	0.5967±0.0112

Table 6.5: Depth estimator performance summary table.

6.8. DISCUSSION

In this chapter, the implementation of the proposed 1D-TOF system is described. The main idea is to produce self-contained sub-modules for the system. So, the modular design is able to adapt a more complex 1D-TOF system according to a designer's requirements.

In the listed sections, the system is decomposed into sub-modules, which are presented and validated individually. Each sub-module is designed based on the specifications in table 3.2. Among them, more space is used to describe the hardware implementation of the trained ANN algorithm, to ensure a reliable estimation of depth. The

[12]The estimated depth results for MATLAB are from table 5.5. They are listed for performance comparison.

results are also compared with the simulation results from MATLAB. Although less accurate than the results from the MATLAB, the DE module still works well to recognize these patterns and give valid results.

BIBLIOGRAPHY

[1] Andrea Gallivanoni, Ivan Rech, and Massimo Ghioni. “Progress in Quenching Circuits for Single Photon Avalanche Diodes”. In: *IEEE Transactions on Nuclear Science* 57.6 (2010), pp. 3815–3826. DOI: 10.1109/TNS.2010.2074213.

[2] C. Inacio and D. Ombres. “The DSP decision: fixed point or floating?” In: *IEEE Spectrum* 33.9 (1996), pp. 72–74. DOI: 10.1109/6.535397.

[3] MATLAB dec2bin. *Convert decimal integer to its binary representation, version 9.12.0.1884302 (R2022a)*. (Last accessed: 31/05/2022). The MathWorks Inc., 2022. URL: https://nl.mathworks.com/help/matlab/ref/dec2bin.html?s_tid=doc_ta.

[4] Xilinx. *KC705 Evaluation Board User Guide for the Kintex-7 FPGA*. V1.9. (Last accessed: 31/05/2022). AMD. Feb. 2019. URL: https://docs.xilinx.com/v/u/en-US/ug810_KC705_Eval_Bd.

7

CONCLUSION

7.1. INTRODUCTION

The presented work described the system level design, and hardware implementation of a 1D-TOF LIDAR system. In this chapter, the achievements of this work are provided. Also, the recommendations for future works in the field of 1D-TOF LIDAR ranging is discussed in section 7.3.

7.2. ACHIEVEMENTS

As stated in the beginning, the goal of this work is to develop a 1D-TOF system for distance ranging, in the context of consumer electronics applications. Based on the goal, the work was divided into system modeling, algorithm design and hardware implementation stages. In the following sections, the achievements obtained from each chapter are provided.

7.2.1. SYSTEM MODELING

In chapter 3, a novel approach of optical modeling of D-TOF LIDAR is proposed. This approach is aimed to be applicable to any D-TOF LIDAR context and it is not limited to short distance ranging only. And, it is able to simulate photon timestamp generation, in order to obtain time-of-flight (TOF) histograms.

This optical model is developed for arbitrary D-TOF LIDAR scenarios. The key idea of the modeling approach is to use a pixellated target surface and a pseudo-sequential ray tracing algorithm, to avoid strong assumptions [1, 2]. In the model, the target is divided into sub-elements. And, the pseudo-sequential ray tracing algorithm is utilized in every sub-element to trace rays between the laser source and the target, and between the target and the SPAD array. For example, the light reflected back from the target is divided into point diffuse sources, in which each source is modeled as a standalone perfect light diffuser.

The photon processing flow follows after the optical model, in which the photons

from signal and noise follow specific types of probability density functions. The time-stamps of the photons are then recorded to build a TOF histogram.

This novel approach overcomes the disadvantage that traditional D-TOF LIDAR model gives inaccurate values at a very close distance, which let the D-TOF optical model be further refined.

7.2.2. TRADE-OFF ANALYSIS

The approach to evaluate system trade-offs enables us to observe the dependency of the system parameters and how they impact its performance. The proposed approach contains three analyses: depth range, depth resolution and failure rate. They serve as the tools which can visualize the non-linear effects. Moreover, the results from trade-off analysis can be used to find an optimal parameter set for a 1D-TOF LIDAR system, given a specific set of requirements.

A major contribution is, that the trade-off analysis provides a methodology about the non-linearity study in TOF histograms. It gives an intuitive observation on the dependencies, which helps designers to adjust their system parameter selections.

7.2.3. DEPTH ESTIMATION ALGORITHM

The achievement of this chapter is the proposed application of an ANN algorithm in 1D-TOF system. In chapter 5, we present three algorithms, including peak detection, noise rejection and ANN algorithm. They focus on finding the TOF in the histogram and filtering the noise. By comparing the performance of each of them, we conclude that the ANN algorithm appears to be the most effective one to identify the depth. The analysis in this chapter gives simulation results that meet the design requirements, with respect to the accuracy and robustness of the algorithm.

7.2.4. HARDWARE IMPLEMENTATION

In chapter 6, the implementation of the proposed 1D-TOF LIDAR system is described. One contribution is that the system components are designed to be modular blocks, which can be used for a more complex system without any major changes in future hardware implementations. Also, we examine the performance of every sub-module, to give an independent validation analysis. We described in detail the trained ANN algorithm, which gives accurate results after implementation on the FPGA.

7.3. OUTLOOK AND FUTURE WORK

During the development of this work, there were some interesting observations that can serve as recommendations for the future works. And, they are the followings:

- The optical model can be further improved, in order to model some environmental conditions such as rain, foggy weather. This is an essential step for an automotive LIDAR, in which occlusion penetration is the most challenging obstacle.
- In all simulations, the target is assumed as a 0.26×0.2 m^2 rectangle. However, we believe that the shape of the target plays a role in the LIDAR detection. So, we propose to verify the ANN performance for different target shapes.

- In Fig. 5.1, the counts are marked in two colors, in which the blue counts are from noise timestamping and the orange counts are from signal timestamping. Two effects can be found. First, the noise photons show an exponential shape, but plenty of spiny protrusions on curves are observed. They might interfere with the signal peak detection. Moreover, it can be found that there is an overlapped area for both photons, which could aggravate the peak-shift effect of the system non-linearity. The ANN was able to perform an accurate depth estimation regardless of these two effects. However, the development of an analytical model, which includes these effects, would contribute to gain a deeper understanding of their causes.

From these observations and considerations, we believe that 1D-TOF LIDAR system is an interesting topic in the future for target ranging. We believe that the neural network estimation methods will become popular in 1D-TOF LIDAR systems. In particular, we found that the ANN algorithm is a robust and efficient depth estimation method. However, a question arises that, what will be the final limitation of the neural network estimation method, the accuracy, the resources utilization, or the network complexity?

ACKNOWLEDGEMENTS

This is the final report of my thesis project, which started in August, 2021 and ends in August, 2022. During this year, I have received a lot of help and support from many people. And I would like to express my appreciation to those who helped and supported me.

First I would like to express my special thank to my supervisor Albert Theuwissen and Padmakumar Rao for bringing me guidance and providing me with suitable arrangements during this period. And, I would also say thanks to my daily co-supervisor Esteban Venialgo Araujo for guiding me during this long journey. This project would not have been developed smoothly without your support and guidance.

I would also like to thank people in Silicon Integrated B.V. Especially, I want to thank my company supervisor Kezheng Ma for guiding me into the digital IC designing field, and Ting Gong for helping me understand TOF system design.

Next, I would say thanks to Ignacio Garcia Ezquerro, Jian Sun, Jingyuan Bai, and Quanhao Yu for sharing experiences and joy with me during my study. And thanks to Ruijun Zheng and Leonardo Johannes In't Veen for taking care of my cat sticky tofu.

Finally, I want to thank my parents for giving me full support to let me study abroad and make me who I am. And I also want to express my thanks to my girlfriend Mingyue Yuan, for letting me feel beloved.

CURRICULUM VITÆ

Mingzhe CHEN

03-04-1998	Born in Qinhuangdao, Hebei, China.

EDUCATION

2020-2022	Technische Universiteit Delft Master of Science Microelectronics
2016-2020	Sichuan University Bachelor degree Microelectronics
2018	Technische Universität Berlin Exchange program Computational Engineering

EXPERIENCE

2021-Present	Digital IC Design Engineer Internship & MSc thesis project Silicon Integrated B.V.
2019-2020	Assistant Instructor Part-time College of Physics, Sichuan University