



TOWARDS HIGHLY EFFICIENT ALGORITHMS
AND HARDWARE ARCHITECTURE DESIGN
FOR SINGLE-PHOTON SIGNAL PROCESSING

Zhenya Zang

Strathclyde Institute of Pharmacy and Biomedical
Sciences

University of Strathclyde

A thesis submitted for the degree of

Doctor of Philosophy

Supervisor:

Dr. David Day Uei Li

January 14, 2025

Copyright

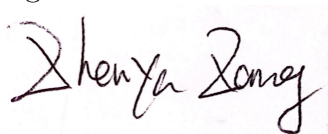
This thesis is submitted to the University of Strathclyde for the degree of Doctor of Philosophy in the Faculty of Science.

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by the University of Strathclyde Regulation 3.51. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Date: January 14, 2025

Signed:

A handwritten signature in black ink, appearing to read 'Zhenya Zong', is written over a light pink rectangular background.

Abstract

In recent decades, single-photon detectors have emerged as crucial technologies for 3D remote sensing and biomedical applications. However, extracting essential parameters from encoded single-photon data presents significant challenges due to the ill-posed nature of parameter reconstruction, leading to high computational complexity. Robust yet compact algorithms are essential to ensure accuracy and computational efficiency. Furthermore, implementing these efficient algorithms on reconfigurable hardware processors promotes portability and real-world applicability.

This thesis addresses these challenges through three interrelated topics that integrate signal processing on single-photon data, deep learning, and hardware implementation. For each topic, quantitative comparisons are conducted between our algorithms and state-of-the-art methods, demonstrating the superiority of our compact algorithms and hardware architectures.

In the first topic, depth images are reconstructed from 3D point cloud data captured by a single-photon avalanche diode (SPAD) array, even under extreme low signal-to-background ratios (0.2, 0.04, and 0.02) per pixel. A low-bit quantization strategy is applied to the 3D DL model to achieve a small model size while maintaining accuracy.

The second topic focuses on fluorescence lifetime reconstruction of both synthetic data and real data from experiments, utilizing 1D temporal point spread functions (TPSF) acquired by a photomultiplier tube (PMT) coupled with a time-correlated single-photon counting (TCSPC) system. A lightweight 1D DL model, in conjunction with TPSF compression and a customized hardware processor, facilitates rapid and accurate lifetime image reconstruction. This approach surpasses conventional DL models and non-linear fitting methods in performance.

In the third topic, we accurately reconstruct the blood flow index and coherence factor from autocorrelation functions for a diffuse correlation spectroscopy (DCS) system using customized 1D DL. A processor implemented on a reconfigurable device paves the way for integrating portable DCS systems in the future.

Through these contributions, this thesis advances the field of single-photon signal processing by providing compact algorithms and hardware implementations that improve accuracy, computational efficiency, and portability in single-photon applications for 3D sensing and biomedical imaging.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor, Dr. David Day Uei Li, for giving me the opportunity to pursue my Ph.D. studies. This thesis could not have been completed without his patient supervision during technical discussions and paper revisions. With sufficient flexibility, I was able to delve into my favorite research areas. Moreover, I am grateful that he helped me apply for a scholarship, which remarkably assisted me in focusing on my research. Beyond research, our group outings marked unforgettable and enjoyable times throughout my study.

I would also like to express my gratitude to DataLab and Photon Force Ltd. for their research funding, which covered my living expenses. I learned imperative skills for a Ph.D. student, such as academic presentation and writing, from the training sessions provided by DataLab. Additionally, I want to appreciate the support from all group members, not only for research discussions but also for pleasant times on basketball courts, in lounges, and during video games.

Last but not least, I want to express my love and gratitude to my parents. They always try their best to provide me with the best educational resources and living conditions without any complaints.

List of Symbols

- α_i : Amplitude Fractions of the i^{th} Lifetime Component
- α : Ratio between Dynamic Scatterers and All Scatterers
- β : Coherence Factor
- β_h : Hyper-parameter for Total Variance
- β_{ELM} : Learned Parameters in ELM
- $\beta(i)$: Learned Parameter in Batch Normalization of the i^{th} output channel
- δ : Accuracy Threshold
- δ_τ : Noise of Autocorrelation Function in DCS
- δ_{BN} : Statistical Standard Deviation of Input Feature Map
- ϵ : Poisson Noise
- η : Quantum Efficiency
- γ : Attenuation Factor
- $\gamma(i)$: Learned Parameter in Batch Normalization of the i^{th} output channel
- θ : Trained Parameters in Deep Neural Network
- λ : Wavelength
- λ_{ELM} : Hyper-parameter in ELM Training
- μ_a : Absorption Coefficient
- μ_{BN} : Statistical Mean of Input Feature Map
- μ'_s : Reduced Scattering Coefficient

ρ : Source-Detector Distances

τ : Lag Time in DCS

τ_A : Amplitude-Weighted Average Lifetime in FLIM

τ_I : Intensity-Weighted Average Lifetime in FLIM

Γ : Decay Rate of Electric Field Autocorrelation Function

\mathcal{L} : Loss Function

\mathbf{Y} : Ground Truth Data for Training ELM

List of Abbreviations

AC	Adder-Convolutions
ACF	Autocorrelation Function
ACNN	Adder-Based Convolutional Neural Network
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuit
ALR	Adaptive Learning Rates
AXI	Advanced eXtensive Interface
BCMM	Bi-Decay Centre of Mass Method
BN	Batch Normalization
BFi	Blood Flow Index
BRAM	Block Random Access Memory
CDE	Correlation Diffusion Equation
CMM	Centre of Mass Method
CNN	Convolutional Neural Network
CW	Continuos Wave
CS	Compress Sensing
CPS	Count per Second
DCS	Diffuse Correlation Spectroscope
DDR	Double-Data-Rate Synchronous DRAM

DFF D-flip-flop
DSP Digital Signal Processing
DNN Deep Neural Network
ELM Extreme Learning Machine
FM Feature Map
FRET Förster Resonance Energy Transfer
FLIM Fluorescence Lifetime Imaging
FLAN Fluorescence Lifetime AdderNet
FLOPS Floating-Point Operations per Second
FPGA Field Programmable Gate Array
FLP Floating Point
FWHM Full Width at Half Maximum
FXP Fixed Point
GNR Gold Nanorod
GPU Graphics Processing Unit
GT Ground Truth
HLS High-Level Synthesis
KL Kullback-Leibler
IRF Instrument Response Function
KL Kullback-Leibler
LiDAR Light Detection and Ranging
LM Log-Matched
LS logarithmic Scale
LSTM Long Short-Term Memory
LUT Look-up Table
MAE Mean Absolute Error

MC Monte Carlo
MCX Monte Carlo eXtreme
MLE Maximum Likelihood Estimation
MLP Multilayer Perceptron
MSE Mean Square Error
MPSoC Multiprocessor Systems-on-Chip
NLSF Non-Linear Square Fitting
PDF Probability Density Function
PE Processing Element
PMT PhotoMultiplier Tube
PS Processing System
PL Programmable Logic
PPMS Pixel per Millisecond **PMS** Pixels per MilliSecond
SBR System Background Ratio
SDD Source-Detector Distance
SGD Stochastic Gradient Descent
SNR System Noise Ratio
SIMD Single Instruction Multiple Data
SSIM Structural Similarity Index
SoC System on Chip
SPAD Single-Photon Avalanche Diode
SLFN Single hidden Layer Feed-forward Neural network
TCSPC Time-Correlated Single-Photon Counting
ToA Time of Arrival
ToF Time of Flight
TDC Time to Digital Converter

TPSF Temporal Point Spread Function

TV Total Variation

UAC Unified Added-Convolution

VFPU Vector Floating-Point Unit

VM Vector-Multiplication

Publication List

- **Z. Zang** and D. D. U. Li, "Object Classification through Heterogeneous Fog with a Fast Data-driven Algorithm Using a Low-Cost Single-Photon Avalanche Diode Array," *Optics Express*, vol. 32, no. 19, pp. 33294-33304, 2024.
- **Z. Zang**, Q. Wang, and D. D. U. Li, "Towards High-performance Deep Learning Architecture and FPGA Accelerator Design for Robust Parameter Reconstruction of Diffuse Correlation Spectroscopy," *Computer Methods and Programs in Biomedicine*, 108471, 2024.
- Q. Wang, **Z. Zang**, and D.D.U. Li, "Quantification of Blood Flow Index in Diffuse Correlation Spectroscopy Using a Robust Deep Learning Method," accepted, *Journal of Biomedical Optics*, vol. 29, no. 1, 015004, 2024.
- **Z. Zang**, D. Xiao, Q. Wang, Z. Jiao, Y. Chen, and D.D.U. Li, "Compact and Robust Deep Learning Architecture for Fluorescence Lifetime Imaging and FPGA Implementation," *Methods and Applications in Fluorescence*, vol. 11, no. 2, 025002, 2023.

-
- **Z. Zang**, U. Dolinsky, P. Ghiglio, S. Cherubin, M. Goli, and S. Yang, "Building a Reusable and Extensible Automatic Compiler Infrastructure for Reconfigurable Devices," pp. 351-352. *33rd International Conference on Field-Programmable Logic and Applications (FPL)*, 2023.
 - Z. Jiao, **Z. Zang**, Q. Wang, Y. Chen, D. Xiao, and D. D. U. Li, "PAIM (π M): Portable AI-enhanced Fluorescence Microscope for Real-Time Target Detection," *Optics and Laser Technology*, vol. 163, 109356, 2023.
 - **Z. Zang**, D. Xiao, Q. Wang, W. Xie, Y. Chen, and D. D. U. Li, "Fast Analysis of Time-Domain Fluorescence Lifetime Imaging via Extreme Learning Machine," *Sensors*, vol. 22, no. 10, 3758, 2022.
 - Q. Wang, Yahui Li, D. Xiao, **Z. Zang**, Z. Jiao, and D. D. U. Li, "Simple and Robust Deep Learning Approach for Fast Fluorescence Lifetime Imaging," *Sensors*, Vol. 22, No. 19, 7293, 2022.
 - **Z. Zang**, D. Xiao, Q. Wang, Z. Li, Y. Chen, and D. D. U. Li, "Hardware Inspired Neural Network for Efficient Time-Resolved Biomedical Imaging," *44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1883-1886, 2022.
 - D. Xiao, **Z. Zang**, Q. Wang, Z. Jiao, F. Rocca, Y. Chen, and D. D. U. Li, "Smart Wide-Field Fluorescence Lifetime Imaging System with CMOS Single-Photon Avalanche Diode Arrays," *44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1887-1890, 2022.

-
- D. Xiao, **Z. Zang**, W. Xie, N. Sapermsap, Y. Chen, and D. D. U. Li, "Spatial Resolution Improved Fluorescence Lifetime Imaging via Deep Learning," *Optics Express*, vol. 30, no. 7, pp. 11479-11494, 2022.
 - **Z. Zang**, D. Xiao, and D. D. U. Li, "Non-Fusion Time-Resolved Depth Image Reconstruction Using a Highly Efficient Neural Network Architecture," *Optics Express*, vol. 29, no. 13, pp. 19278-19291, 2021.
 - D. Xiao, **Z. Zang**, N. Sapermsap, Q. Wang, W. Xie, Y. Chen, and D. D. U. Li, "Dynamic Fluorescence Lifetime Sensing with CMOS Single-Photon Avalanche Diode Arrays and Deep Learning Processors," *Biomedical Optics Express*, pp. vol. 12, no. 6, pp. 3450-3462, 2021.
 - W. Xie, H. Chen, **Z. Zang**, D. D. U. Li, "Multi-Channel High-Linearity Time-to-Digital Converters in 20 nm and 28 nm FPGAs for LiDAR Applications," *6th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP)*, pp. 1-4, 2020.

Contents

Copyright	i
Abstract	ii
Acknowledgements	iv
List of Symbols	v
List of Abbreviations	vii
Publication List	xi
Contents	xiv
List of Figures	xviii
List of Tables	xxv
1 Introduction	1
1.1 Thesis Structure	4
2 Non-Fusion Deep Learning Framework for Accurate Depth Re- construction for Single-Photon LiDAR	9

CONTENTS

2.1	Background	9
2.2	Prior Work	10
2.3	Problem Definition	12
2.4	Non-Fusion ToA Denoising Model	14
2.4.1	Neural Network Architecture	14
2.4.2	Training Detail	18
2.5	Evaluation	20
2.5.1	Loss Evaluation	20
2.5.2	Synthetic Data	22
2.6	Captured Data	27
2.7	Discussion	29
2.8	Summary and Future Work	33
3	Compact and Robust Deep Learning Architecture for Fluores-	
	cence Lifetime Imaging and FPGA Implementation	34
3.1	Background	34
3.2	Prior Work	36
3.2.1	Deep Learning for FLIM	36
3.2.2	Hardware Platforms for FLIM	37
3.3	Problem Definition	38
3.4	Deep Learning Network Details	39
3.4.1	Network Architecture	40
3.4.2	Preparation for Hybrid Training Data	44
3.5	Training Details	46
3.6	Data Compression Strategies	47
3.6.1	Log-Scale Mapping for Time-Bin	47

CONTENTS

3.6.2	On-Chip Linear Quantization	50
3.7	Synthetic Data Evaluation	50
3.8	Real-Case Study: Fluorescent Beads Discrimination	54
3.8.1	Sample Preparation	54
3.8.2	Optical Setup	55
3.8.3	Quantitative Analysis	55
3.9	Hardware Implementation	60
3.10	Summary and Future Work	66
4	Fast Fluorescence Lifetime Imaging Analysis Using Extreme Learning Machine for Time-Domain Single-Photon Detector	68
4.1	Background	68
4.2	Apply ELM to FLIM	69
4.2.1	ELM Theory	70
4.3	ELM Network Architecture	71
4.4	Synthetic Data Analysis	72
4.4.1	Comparisons of Individual Lifetime Components	73
4.4.2	Comparisons of τ_A	78
4.4.3	Comparisons of τ_I	80
4.5	Experimental FLIM Data Analysis	81
4.5.1	Experimental Setup and Sample Preparation	82
4.5.2	Algorithm Evaluation	83
4.5.3	Low Counts Scenarios	85
4.6	Summary and Future Work	86

CONTENTS

5	Towards High-performance Deep Learning Architecture and Hardware Accelerator Design for Robust Parameters Analysis in Diffuse Correlation Spectroscopy	88
5.1	Background	88
5.2	Prior Work	91
5.2.1	Algorithms Review	91
5.2.2	Hardware Processor Review	92
5.3	Analytical Model	93
5.3.1	Monte Carlo Simulation	98
5.4	DL Architecture	98
5.4.1	Accuracy Evaluation	102
5.4.2	Accelerator Architecture	104
5.4.3	Accelerator Evaluation	107
5.5	Summary and Future Work	109
6	Conclusions and Final Remarks	110
6.1	Summary of Remarks	110
6.2	Potential Future Works	111
	References	114

List of Figures

1.1	The structure of the thesis.	5
2.1	Time intervals between emitted and detected photons are measured and digitalized by the TCSPC system. Detected photons are accumulated to generate a 3D tensor. Each data cube (blue) is a histogram, and neural network post-processing is adopted to retrieve the time-bin index representing the average distance. . . .	12
2.2	The proposed model's architecture. The input is a ToA tensor corrupted by background noise. By adopting dense connections in each row, short- and long-range information can be fully explored. A more robust depth map can, therefore, be obtained.	15
2.3	Dataflow and quantization process in each convolutional layer. In the forward propagation phase, the weight and activation are quantized with a low-bit accordingly.	17
2.4	(a) and (b). Training and validation loss; (c) and (d). RMSE plots with different quantization cases. Each plot contains the original FLP 32-bit format and three quantization cases.	21

LIST OF FIGURES

2.5	Depth images reconstructed from the ToA tensor. The fidelity differences between monocular-SPAD fusion and the proposed model are marked in red boxes. The proposed model can reveal more details for long-distance objects and obtain a lower overall RMSE when SBR equals 0.04.	23
2.6	Reconstructed images using different algorithms. The SBRs from the first row to the last row are 0.2 (2 target photons and 10 background photons), 0.04 (2 target photons and 50 background photons), and 0.02 (2 target photons and 100 background photons). Our model can obtain lower RMSE than other algorithms when SBR equals 0.2 and 0.04 and achieves a comparable RMSE with <i>Sun et al.</i> , even without fusion strategies.	25
2.7	(a). Intensity images of the five scenes used for fusing with Lindell <i>et al.</i> 's and Sun <i>et al.</i> 's models. (b). Detected ToA data cubes of five scenes. The bouncing ball and a hallway (the 4 th and 5 th rows) scenes contain intense ambient light, leading to blurred impact. (c). Reconstructed depth maps. Lindell <i>et al.</i> used the intensity-SPAD fusion, and Sun <i>et al.</i> used the monocular-SPAD fusion. The processing time is proportional to the network's depth, and the RMSE decreases for a deeper network.	28
2.8	(a) RMSE and (b) squared relative difference (Sq rel) plots in terms of SBR. The dashed lines indicate neural network-based algorithms, and the rest represent optimization based algorithms. 'Sun' and 'Lindell w/' shown in the legend use the monocular-SPAD and intensity-SPAD fusion strategy, respectively.	30

LIST OF FIGURES

2.9	The processing time is proportional to the network's depth, and the RMSE decreases for a deeper network.	32
3.1	Overview of FLAN architecture. Histograms from each pixel will be processed consecutively, and the pixels from the background will be discarded initially. Prostatic cells coated with biomarkers are adopted here for illustration.	41
3.2	Procedures for generating mono- and bi-exponential TPSFs. The simulated IRF $I(t)$ convolved with mono- and bi-exponential PDFs $h(t)$. After that, peak photon counts N_p were applied to mimic real histograms. And then Poisson noise $\epsilon(t)$ was added at the final step.	45
3.3	Training and validation loss curves from 1D CNN, FLAN, and FLAN + LS.	46
3.4	IRFs, original and compressed synthetic decays.(a) Mono-exponential decay with lifetime. (b) Bi-exponential decay with lifetime. . . .	48
3.5	Nonlinear time-bin mapping, merging original 256 time-bin to compressed 80 time-bin.	48
3.6	Synthetic GT and reconstructed τ_A and τ_I FLIM images. MSE is used to evaluate the accuracy. (a),(b), and (c) reconstructed τ_A in [1000,5000], [100,1000], and [10,100] photon-count. (d),(e), and (f) reconstructed τ_I in [1000,5000], [100,1000], and [10,100] photon-count.	52
3.7	Lifetime distributions of τ_A in (a),(b), and (c); τ_I in (d),(e), and (f) retrieved from different algorithms in different levels of photon count.	53

LIST OF FIGURES

3.8	Phasor plots of unmixed (Yellow-green) fluorescent beads and mixed (Yellow-green and Crimson), showing (a) one cluster and (b) two clusters.	56
3.9	Unmixed beads (Yellow-green) evaluation using accuracy and precision	57
3.10	Lifetime reconstruction of mixed beads (Yellow-green and Crimson).	57
3.11	Reconstructed lifetime distributions of mixed beads using different algorithms. The reference lifetimes of Yellow-green and Crimson are 2.1 ns and 3 ns, respectively.	58
3.12	Reconstructed lifetime images of <i>Convallaria majalis</i> cells.(a) intensity image; (b) phasor projection image; (c) and (d) reconstructed τ_A and τ_I from FLAN; (e) and (f) reconstructed τ_A and τ_I from FLAN+LS.	59
3.13	Pipelines of files generation for (a) training on PC and (b) inference on FPGA.	61
3.14	Overview of the FPGA hardware platform embedding four FLANs.	61
3.15	Architecture of UAC. BRAMs storing and caching learned parameters and FMs are partitioned to feed data to corresponding ACs. Multiple ACs and BN modules are instantiated to improve parallelism. (b) BRAMs were partitioned into smaller portions to cache parallel data on-the-fly.	62
3.16	Relationships between batch sizes and throughput (pixel/ms) in the inference phase on CPU, GPU, and FPGA + CPU.	65

LIST OF FIGURES

4.1	ELM is used for lifetime analysis. The input data are a 1-D pixel-wise histogram from the raw point cloud that contains 256 time bins. The histogram is fed into a single-hidden-layer ELM, and lifetime parameters (τ_1 , τ_2 , and α) can be obtained from output nodes.	72
4.2	Box plots of absolute error versus different peak intensity levels regarding testing datasets. (a,b) Single lifetime estimations of mono-exponential decays from ELM and NLSF, respectively. (c,d) Double lifetime estimations of bi-exponential decays from ELM and NLSF, respectively. (e,f) τ_A estimated by ELM and NLSF, respectively.	74
4.3	Lifetime parameters' estimation results, photon counts for each pixel were randomly picked between 25 and 500. (a) The estimated single lifetime using a mono-exponential decay model, where $\tau \in [0.1, 5]$ ns from top to down in the image. (b) The two estimated lifetimes using a bi-exponential decay model where $\tau_1=0.3$ ns, $\tau_2=3$ ns, and $\alpha \in [0, 1]$, from the top down. (c) Two phasor plots of GT distributions of (a, b). (d) Prediction accuracy and R^2 of τ_A from ELM and NLSF, with $\tau_1=0.3$ ns, and $\tau_2=2.5$ ns, respectively. . . .	75
4.4	F -value ys with τ_1 , τ_2 , and α in the ranges $[0.1, 1]$ ns, $[1, 3]$ ns, and $[0, 1]$, respectively. (c,d) Bias per histogram for mono- and bi-exponential decays.	77
4.5	(a) Intensity image of GT τ_A in n exact ranges. $I_p c$ depicts total photon counts in one pixel. The range from 40 to 400 is viewed as low photon counts. (b) the GT τ_A lifetime image with the range $[0.3, 2.5]$ ns. (c-e), τ_A images from ELM, NLSF, and BCMM. . . .	77

LIST OF FIGURES

4.6	(a) GT τ_I image in exact ranges. (b-d), Reconstructed τ_I images from ELM, NLSF, and CMM for bi-exponential decays.	79
4.7	(a), (b) Loss curves and time consumption vs. different numbers of nodes in the hidden layer.	79
4.8	Lifetime analysis of prostatic cells loaded with gold nanoprobe. (a) The intensity image, b) phasor plot, and (c) phasor projection image. (d-g) τ_A restored by ELM, 1D CNN, NLSF, and CMM. (h) Lifetime histograms of ELM, 1-D CNN, NLSF, and BCMM. .	83
4.9	(a) Intensity images with different scales of colorbars, scanning cycles were set to 10, 40, 60, and 80. Colorbars are unified. (b) τ_A ages and pixel occurrence reconstructed by ELM in different cycles.	86
5.1	The concept of spatial diffuse reflectometry in a semi-infinite geometry.	94
5.2	$g_2(\tau)$ curves with fixed optical parameters but with (a) different averaging time ($t = 1$ seconds, 5 seconds, and 10 seconds) and (b) with different photon intensities ($I = 20$ kcps, 40 kcps, and 80 kcps).	96
5.3	Comparison between MC simulation and analytic model for $g_1(\tau)$ and $g_2(\tau)$ of milk. (a) and (b) generated and fitted curves using MCX and analytical models. (c) and (d) Euclidian distance between simulated and fitted curves.	97
5.4	ACNN architecture in training and inference phases.	99
5.5	Training and validation loss curves of ACNN and CNN. (a) and (b), ACNN training and validation loss curves in 109 epochs. (c) and (d), CNN training and validation loss curves in 96 epochs. . .	100

LIST OF FIGURES

5.6	R -square and MSE evaluation between ACNN, CNN, NLSF. (a) and (b) R -square of BFi and from ACNN. (c) and (d) R -square of BFi and β from CNN. (e) and (f) R -square of BFi and β from NLSF.	101
5.7	Histograms depict the weights distribution of CNN and ACNN. (a) and (b) represent Gaussian and Laplace distributions of CNN's and ACNN's weights.	103
5.8	Accuracy evaluation of ACNN and CNN under different levels of photon rate in the noise model. (a) and (b) reconstructed β and BFi.	103
5.9	Hardware architecture integrates intensity temporal ACF computation and ACNN accelerators (10 cores on Zynq and 15 on Zynq UltraScale+). (a). The architecture overview illustrates the data transfer and functionalities of each module. (b). Detailed architecture of each DL core, data path and memory access were depicted with back and yellow arrows, respectively. (c). Detailed structures of each UAC illustrate parallelism in the input channel, output channel, and kernel size.	106
5.10	DL cores implementation on Zynq-7000 and Zynq-UltraScale+ with different quantization FXP schemes, and their corresponding MAE of BFi and β .	108

List of Tables

2.1	Compression rates of our FLP model and existing networks in terms of the parameter size, training time and compression rate. .	22
2.2	Extensive evaluations between Sun <i>et al.</i> 's [23] and the proposed models with $SBR = 0.04$. The proposed model achieves more robust results across five evaluations metrics.	25
2.3	Quantitative analysis of the proposed and existing algorithms over seven indoor scenes. Three tables are for three different SBR levels - 0.2, 0.04, and 0.02. And the underlined numbers mean that they are comparable to the best existing results.	26
3.1	Comparisons of DL Architectures for FLIM	43
3.2	Ranges of lifetime parameters for synthetic data.	46
3.3	Comparisons of Hardware-Efficient DL Networks Implemented on FPGA.	64
4.1	Time Consumption (Seconds) of NLSF and ELM for Reconstructing Lifetime Parameters.	78
4.2	Comparisons of Existing NN Architecture for Lifetime Estimation.	80
5.1	Detailed computational information of each layer.	100

LIST OF TABLES

5.2	Evaluation results of ACNN accelerators with different quantization bit-width on Zynq-7000 and Zynq-UltraScale+ FPGA.	106
5.3	Performance comparisons of CPU, GPU, and FPGA-based SoC when processing different numbers of pixels (ACFs) for each batch.	107

Chapter 1

Introduction

Many time-domain active imaging systems that utilize single-photon detectors and lasers, such as single-photon LiDAR [1, 2], fluorescence lifetime imaging (FLIM) [3, 4], and diffuse correlation spectroscopy (DCS) [5, 6], can be effectively described using corresponding mathematical models. These models capture the optical interactions between lasers, objects, and detectors. For instance, temporal point spread functions (TPSFs) or histograms generated from collected photons in LiDAR and FLIM are modeled based on the experimental setup and key parameters of interest, such as time-of-flight (ToF) and fluorescence lifetime, respectively. In the case of DCS, autocorrelation functions, calculated directly from time-resolved photon intensity, can be derived using an analytical model involving the blood flow index, which characterizes the transport and scattering behavior of photons in tissue or phantoms. These models typically integrate optical illumination behavior, observed data, and a function containing the parameters to be reconstructed. However, reconstructing these parameters from the function using observed data constitutes an inverse problem, posing significant challenges due to its inherently ill-posed nature.

Persistent issues such as low signal-noise ratios (SNR), and slow computational speeds have continued to impede real-time and accurate parameter reconstruction, despite the use of typical optimization-based or fitting strategies tailored to specific inverse problems [7–11]. To tackle these challenges, there is a growing interest in leveraging mathematical models to generate authentic synthetic data or developing automated data collection pipelines. Deep neural networks (DNNs) have emerged as promising solutions to address these challenges in a data-driven manner. Additionally, with the non-iteration properties of DNNs’ feed-forwards inference and compressible raw data from single-photon sensors, compression strategies and hardware acceleration can be achieved.

In this thesis, we aim to tackle mentioned challenges within the realm of emerging photonic signal processing topics using data-driven methods.

- **Chapter 2: photon efficient DNN and data compression for data-driven 3D ranging:** Single-photon avalanche diode (SPAD) arrays are emerging detectors for modern time-resolved single-photon LiDAR systems. The ToF of individual photons, emitted from the light source and subsequently reflected from the object, is encoded into histograms, delineating the distribution of the photons’ ToF for each pixel. Crucially, the reconstruction of the object’s geometry relies on accurately retrieving the peak from these histograms. Nonetheless, this task is often hindered by the presence of low signal-to-background ratio (SBR). We endeavor to tackle the challenges posed by low SBR through the utilization of a high computational efficiency data-driven approach. This chapter was derived from the published paper:

Z. Zang, D. Xiao, and D. D. U. Li, "Non-Fusion Time-Resolved Depth Image Reconstruction Using a Highly Efficient Neural Network Architecture," *Optics Express*, vol. 29, no. 13, pp. 19278-19291, 2021.

- **Chapter 3: Hardware-friendly DNN and hardware implementation for FLIM:** While fluorescence lifetime reconstruction is commonly performed offline on CPUs or GPUs, often exhibiting low computational efficiency, we propose a hardware-friendly data-driven framework and an FPGA-based embedded solution to accelerate this process. Furthermore, we tackle the challenge of low photon count conditions, especially prevalent in specialized bio-tissues like the retina, while ensuring high accuracy in lifetime estimation. This chapter was derived from the published paper:

Z. Zang, D. Xiao, Q. Wang, Z. Jiao, Y. Chen, and D.D.U. Li, "Compact and Robust Deep Learning Architecture for Fluorescence Lifetime Imaging and FPGA Implementation," *Methods and Applications in Fluorescence*, vol. 11, no. 2, 025002, 2023.

- **Chapter 4: Fast training and inference for data-driven FLIM:** FLIM is a powerful indicator of molecular dynamics. Emitted individual photons from the stimulated tissue or cells by the light source are collected by a time-resolved single-photon sensor. Intensity information and fluorescence lifetime can be reconstructed pixel-by-pixel or globally using histograms obtained during acquisition time. We propose a data-driven algorithm with a fast training approach to accelerate model learning and lifetime estimation, paving the way for clinical and surgery applications. This chapter was derived from the published paper:

Z. Zang, D. Xiao, Q. Wang, W. Xie, Y. Chen, and D. D. U. Li, "Fast Analysis of Time-Domain Fluorescence Lifetime Imaging via Extreme Learning Machine," *Sensors*, vol. 22, no. 10, 3758, 2022.

- **Chapter 5: Compact DNN and hardware implementation for data-driven DCS:** DCS is a non-invasive method for monitoring the blood flow index (BFi), typically employing coherent near-infrared light. This technique relies on detecting speckle patterns generated by the rapid movement of red blood cells within the tissue. By measuring the temporal autocorrelation function (ACF) of photon counts, BFi can be estimated via reconstruction algorithms. We introduce a compact deep learning (DL) architecture and FPGA-based accelerator designed to achieve rapid embedded reconstruction of BFi. This chapter was derived from the paper being peer-reviewed:

Z. Zang, Q. Wang, and D. D. U. Li, "Towards High-performance Deep Learning Architecture and FPGA Accelerator Design for Robust Parameter Reconstruction of Diffuse Correlation Spectroscopy," submitted to *Computer Methods and Programs in Biomedicine*, 2024.

1.1 Thesis Structure

The structure of the thesis is depicted in Figure 1.1. Although they all fall under the purview of single-photon signal processing, each topic is uniquely tailored to address specific applications, underpinned by distinct and discerning design imperatives. The abstract of each chapter are summarized as follows. For conceptual clarity, the background of each topic will be expounded within the confines of each chapter.

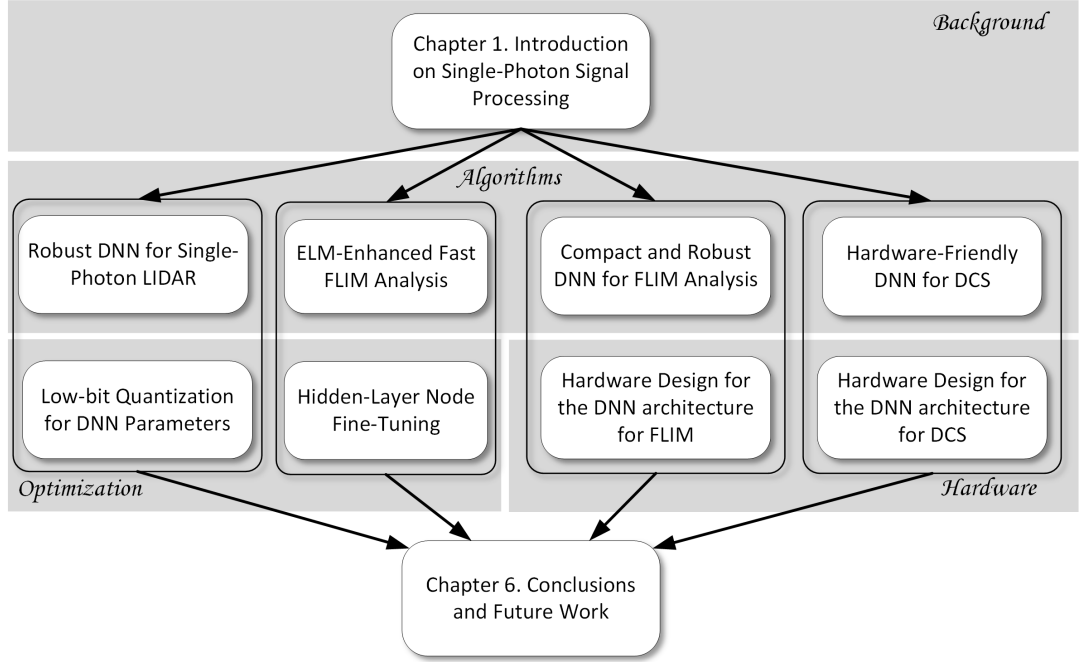


Figure 1.1: The structure of the thesis.

- **Chapter 2:** proposes a photon-efficient, non-fusion DNN that can directly reconstruct high-fidelity depth images from histograms without relying on other guiding images with extremely low SBR - 0.2, 0.04, 0.02. Besides, the DNN is compressed via a low-bit quantization scheme so that it is suitable to be implemented on embedded hardware platforms. The proposed quantized DNN achieves superior reconstruction accuracy and fewer parameters than previously reported networks.
- **Chapter 3:** reports a bespoke adder-based DNN for FLIM analysis. By leveraging the l_1 -norm extraction method, an 1D Fluorescence Lifetime AdderNet (FLAN) without multiplication-based convolutions is proposed to reduce the computational complexity. Further, fluorescence decays are compressed in temporal dimension using a log-scale merging technique to dis-

card redundant temporal information derived as log-scaling FLAN (FLAN+LS). FLAN+LS achieves 0.11 and 0.23 compression ratios compared with FLAN and a conventional 1D convolutional neural network (1D CNN) while maintaining high accuracy in retrieving lifetimes. This chapter extensively evaluates FLAN and FLAN+LS using synthetic and real data. A traditional fitting method and other non-fitting, high-accuracy algorithms are compared with our networks for synthetic data. Our networks attain a minor error in different photon-count scenarios. For real data, fluorescent beads' data acquired by a confocal microscope is presented to validate the effectiveness of real fluorophores, our networks can differentiate beads with different lifetimes. Additionally, FLAN and FLAN+LS are implemented on an FPGA with a post-quantization technique to compress the bit-width, thereby improving computing efficiency. FLAN+LS on hardware achieves the highest computing efficiency compared to 1D CNN and FLAN. This chapter also discusses the applicability of our network and hardware architecture for other time-resolved biomedical applications using photon-efficient, time-resolved sensors.

- **Chapter 4:** presents the extreme learning machine (ELM) for fast and accurate FLIM analysis. This chapter used extensive metrics to evaluate ELM and existing algorithms. First, these algorithms are compared using synthetic datasets. The results indicate that ELM can obtain higher fidelity, even in low-photon conditions. Afterward, ELM is used to retrieve lifetime components from human prostate cancer cells loaded with gold nanosensors, showing that ELM also outperforms the iterative fitting and non-fitting algorithms. By comparing ELM with a computationally efficient DNN,

ELM achieves comparable accuracy with less training and inference time. As there is no back-propagation process for ELM during the training phase, the training speed is much higher than existing DNN approaches. The proposed strategy is promising for edge computing with online training.

- **Chapter 5:** leverages a rigorous analytical model to generate ACFs to train a compact DNN. This chapter assesses the accuracy of the proposed DNN using simulated data based on a single-layer model of milk. Compared to convolutional neural networks (CNN), our lightweight DNN achieves 66.7 % and 18.5 % improvement in mean square error (MSE) for BFi and the coherence factor β , using synthetic data evaluation. The DNN is optimized by using subtraction for feature extraction, considering further hardware implementation. The chapter extensively explores computing parallelism and fixed-point quantization within the DNN. With the DNN's compact size, we employ unrolling and pipelining optimizations for computation-intensive for-loops in the DL model while storing all learned parameters in on-chip BRAMs. We also achieve pixel-wise parallelism, enabling simultaneous, real-time processing of 10 and 15 ACFs on Zynq-7000 and Zynq-UltraScale+ FPGA, respectively. Unlike existing FPGA accelerators that produce BFi and β from ACFs on standalone hardware, our approach is an encapsulated, end-to-end on-chip conversion process from intensity photon data to the temporal intensity ACF and subsequently reconstructing BFi and β . This hardware platform achieves an on-chip solution to replace post-processing and miniaturize modern DCS systems that use single-photon

cameras. While comprehensively comparing the performance running the DNN on CPU, GPU, and FPGA, FPGA surpasses CPU and GPU in terms of computational efficiency.

Chapter 2

Non-Fusion Deep Learning Framework for Accurate Depth Reconstruction for Single-Photon LiDAR

2.1 Background

Depth imaging has been an essential tool in various applications, such as autonomous vehicles [12], vision-guided robotic systems [13], and augmented reality applications [14]. Many strategies have been proposed to obtain depth information from captured intensity images. For example, Zhan *et al.* used machine-learning and monocular depth perception [15] principles to retrieve depth information from RGB images. However, the reconstruction fidelity deteriorates because of the scale ambiguity. The stereo-vision technique mimicking human vision systems [16] is also popular. It uses the triangulation principle to understand spatial information. Conventionally, stereo-based cameras are not photon-sensitive; few reflected photons are detected. Besides, the sensing accuracy of the stereovision approach deteriorates when it works in dark conditions or performs long-distance

measurements, whereas LiDAR can overcome the limitations. LiDAR has become popular in ranging applications. Unlike Radar systems [17] that use radio waves to measure the ToF between transmitted and reflected signals, LiDAR systems adopt pulsed light with a much shorter wavelength to detect an object’s range. Therefore, LiDAR can obtain more accurate spatial information than Radar for seeing objects at a longer distance [1]. SPADs [18] are effective LiDAR sensors due to their singlephoton sensitivity and excellent temporal resolutions. Richardson *et al.* developed and applied low-noise SPAD sensors [19] to time-resolved imaging [3, 20]. Recent advances in silicon manufacturing have introduced more compelling high fill-factor devices [21].

2.2 Prior Work

Convex-optimization [22] and Bayesian inference [23] approaches have been widely adopted to tackle the recovery problem in low photon-flux conditions. Shin *et al.* [7] modelled photon registration behaviors as the rate function [24] of a Poisson process and used the constrained maximum likelihood estimation (MLE) to reconstruct depth images. Later, to improve the reconstruction accuracy, Rapp and Goyal [8] proposed a pixel-wise spatial unmixing strategy to split the signal cluster and background noise. Tachella *et al.* used the Bayesian theory to identify surfaces from pixel-wise histograms [25], using the advanced Markov chain Monte Carlo (MCMC) sampling method to address the maximum-a-posterior (MAP) problem. Although their algorithm yielded outstanding 3D reconstructions with a fast processing speed, some hyperparameters should be adjusted to maintain accuracy and computing efficiency. Quentin *et al* [26] have used the expectation-

maximization (EM) method to estimate multi-spectral and depth profiles. It shows excellent performances in estimating mixed probabilistic models with latent variables.

Deep neural networks (DNNs) feature hierarchical structure, showing powerful learning ability from complex data. Using DNN with sensor-fusion strategies has been a new trend for extracting ToA depth maps. Lindell *et al.* [27] first introduced a U-net neural network [28] merging 2D intensity images and 3D SPAD tensor data to reconstruct depth images. To make the hardware platform more compact, a dedicated neural network was proposed [29] for a SPAD array that can simultaneously generate intensity images and ToA data. Another sensor fusion architecture [30] uses monocular depth perception principles to retrieve a coarse-grain depth map in advance. The corresponding ToA data is fused with the up-projected monocular depth map. This fusion strategy achieved more accurate results than the SPAD-intensity version. A non-local neural network model [31] was introduced to explore the long-range correlation along spatial and temporal dimensions. Zhao *et al.* [32] presents a two-stage network for improving depth estimation in SPAD-based LiDAR 3D imaging, integrating a multi-scale encoder-decoder with a deep boosting network and an intensity-guided edge refinement sub-network to enhance detail and robustness. However, the networks mentioned above have large model sizes and redundant parameters. Therefore, they show a long training time and slow inference speed, not suitable for real-time (10 Hz to 20 Hz or higher frame rate) scenarios. Also, it is intractable to implement them on embedded hardware such as field-programmable gate arrays (FPGA) or application-specific integrated circuits (ASIC).

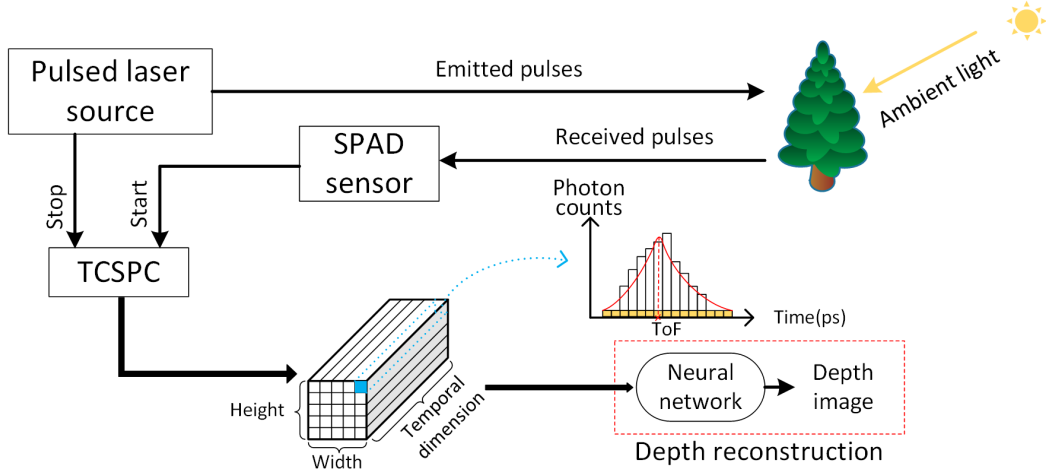


Figure 2.1: Time intervals between emitted and detected photons are measured and digitalized by the TCSPC system. Detected photons are accumulated to generate a 3D tensor. Each data cube (blue) is a histogram, and neural network post-processing is adopted to retrieve the time-bin index representing the average distance.

Figure 2.1 shows a conventional SPAD-based LiDAR system, including a pulsed laser, a SPAD sensor, and a time-correlated single photon counting (TCSPC) module. The TCSPC module includes a picosecond time-to-digital converter (TDC) to measure and time-stamp reflected photons from the target. The system also contains a histogramming module to establish a time of arrival profile for estimating the average depth (or the distance).

2.3 Problem Definition

For raster-scanning and wide-field ranging systems, a laser source generates periodic pulses $s(t)$ to illuminate the target scene. The reflected photon flux can be detected by individual pixels with the quantum efficiency $\eta \in [0, 1)$. We assume that there are only a few detected photons (less than 1 photon per pixel). Therefore, pile-up effects and carriers' crosstalk [33] are negligible. The spatial

resolution of the SPAD array is $(m_1, m_2) \in \{1, 2, \dots, M\}^2$. According to previously published reports [24–27], the number of recorded photons in the time interval $n \in \{1, 2, \dots, N\}$ of each pixel can be formulated as

$$r_{m_1, m_2}[n] = \int_{n\Delta t}^{(n+1)\Delta t} \eta(g * s)(t - \frac{2d_{m_1, m_2}}{c})dt + b_\lambda, \quad (2.1)$$

where g denotes the instrument response function (IRF), Δt is the time bin-width of the TDC, $d \in \mathbb{R}_+^{M \times M}$ is the scene’s depth profile, c is the speed of light, and b_λ is the ambient light with a wavelength λ . The photon arrival behavior can be formulated as an inhomogeneous Poisson process [34]. The dark count [33], triggered by thermally-generated carriers in the SPAD sensor is also considered the time-varying factor in the Poisson process’s rate function. Consequently, the histogram in one pixel with I illumination periods can be formulated by a Poisson process $\mathcal{P}(\cdot)$ with a time-varying arrival function

$$h[t] \sim \mathcal{P}(I((\gamma r[n] + b_d))), \quad (2.2)$$

where the constant γ , represents the attenuation factor caused by photon scattering on the surface and b_d is the dark count rate. Suppose $f(\cdot)$ is the neural network’s feed-forward function, and the input is a noise-corrupted tensor composed of 1D pixel-wise histograms with a 2D spatial resolution M^2 . Therefore, pixel-wise denoised ToA data can be modelled as

$$\hat{h}^{(m_1, m_2)} = f(h^{(m_1, m_2)}; \theta), \quad (2.3)$$

where θ is the parameter set to be learned, and we can use maximum likelihood estimation (MLE) to calculate it as

$$\hat{\theta}_{ML} = \arg \max_{\theta} P(\hat{h}^{(m_1, m_2)} | h^{(m_1, m_2)}; \theta). \quad (2.4)$$

Equation 2.4 can be solved by the neural network’s learning phase to be detailed in the next section.

2.4 Non-Fusion ToA Denoising Model

This section introduces the proposed network architecture, loss function, and low-bit quantization scheme. To conduct a fair comparison with peers’ work, we focus on depth images retrieved in low photon-flux conditions with a low SBR.

2.4.1 Neural Network Architecture

As shown in Figure 2.2, the U-net++ [35] was used as the network’s backbone, modified to a 3D version to denoise the ToA tensor. The network consists of two parts: a main feature extraction module and a refinement module. In Figure 2.2, $x^{i,j}$ represents the nodes where $i \in \{0, 1, 2, \dots, l\}$ is the row number along the down-sampling path (blue arrows) $j \in \{0, 1, 2, \dots, l\}$ is the number of nodes in skipped connections in each row, and l is the level or depth of the down-sampling. The refinement and a differentiable argmax function after the last up-sampled node

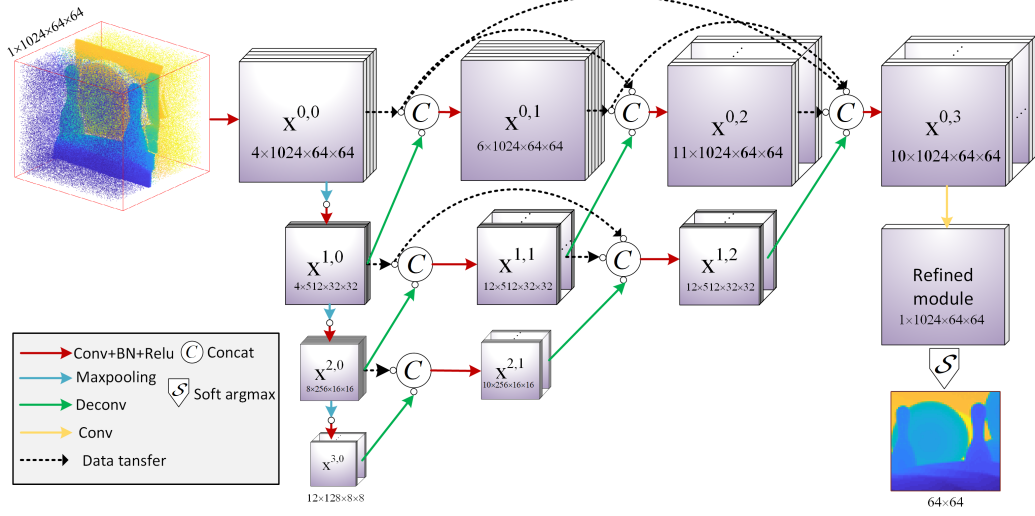


Figure 2.2: The proposed model’s architecture. The input is a ToA tensor corrupted by background noise. By adopting dense connections in each row, short- and long-range information can be fully explored. A more robust depth map can, therefore, be obtained.

are included. The procedure for calculating a feature map can be formulated by

$$x^{i,j} = \begin{cases} \text{Conv}(\mathcal{M}(x^{i-1,j})), & i > 0, j = 0 \\ \text{Conv}([x^{i,k}]_{k=0}^{j-1}, \mathcal{U}(x^{i+1,j-1})), & 0 < j < l \\ \mathcal{S}(\text{Conv}(x^{i,j})) & i = 0, j = l \end{cases} \quad (2.5)$$

where $\text{Conv}(\cdot)$ denotes the convolution operation, $\mathcal{M}(\cdot)$ the max-pooling to perform downsampling, $\mathcal{U}(\cdot)$ the transposed convolution to perform up-sampling, $[\cdot]$ the concatenating layer’s operation, and $\mathcal{S}(\cdot)$ is the soft argmax function to find the bin index.

By adding dense connections in each horizontal node, our architecture can compensate for information lost due to the down-sampling. Besides, long-range and short-range concatenations (black dot lines in Figure 2.2) between horizontal con-

volitional layers can fuse different non-local spatial and temporal information of time-of-arrival (ToA) measurements. Within a pixel, a soft-argmax function [20] is applied to find the index corresponding to the ToF (or the distance) (see Figure 2.2). So the ToA tensor's noise can be censored during the learning phase, and the network generates a squeezed 2D depth map from the denoised ToA tensor. This architecture does not need any guiding images (monocular and intensity) due to the above features, thereby saving processing time and parameters than fusion approaches. The max-pooling as down-sampling operations along the down-sampling path is applied to reduce network parameters and computing time. To make our network portable to embedded hardware, a model compression strategy was applied to simplify it. The bottleneck of embedding neural networks in reconfigurable hardware is due to limited on-chip memory for storing pre-trained parameters and the large memory bandwidth for data transfer. Therefore, we generalize a 2D low-bit parametric quantization scheme [36] for 3D data quantization to compress the model. Multiplication operations of floating-point (FLP) numbers can be converted to bitwise operations of fixed-point (FXP) numbers (in binary). Briefly, suppose x and y are two FXP integers coded with M -bit and K -bit binary digits respectively. The conversions are subject to $x = \sum_{m=0}^M c_m(x)2^m$ and $y = \sum_{k=0}^K c_k(y)2^k$, where $(c_m(x))_m^{M-1} = 0$ and $(c_k(y))_k^{K-1} = 0$ are corresponding binary digits. The dot product of x and y can therefore be indicated by

$$xy = \sum_{m=0}^{M-1} \sum_{k=0}^{K-1} 2^{m+k} \text{bitcount}[\text{and}(c_m(x), c_k(y))], \quad (2.6)$$

where all operations can be performed via economic bitwise operations in FPGAs. More details are provided in [36]. As for our neural network, weights (W) and activation functions' outputs (A) are quantized with different bit-widths,

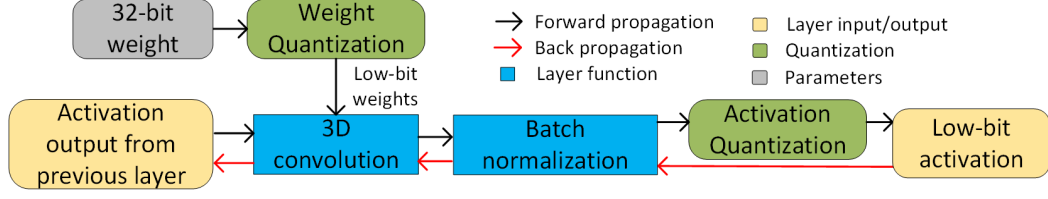


Figure 2.3: Dataflow and quantization process in each convolutional layer. In the forward propagation phase, the weight and activation are quantized with a low-bit accordingly.

denoted as $WXAY$, where X and Y are bit-widths. The quantization process in each convolutional layer is depicted in Figure 2.3. The weights and activations can be quantized with an arbitrary low bit-width during the forward propagation. The weights are firstly normalized via a $\tanh(\cdot)$ function, and a quantization function $\mathcal{Q}_k(\cdot)$ converts the normalized FLP weights to a FXP format, defined in Equation 2.8. To avoid information loss, we keep weight parameters in the first and last layers in the FLP format and compress internal layers' parameters. k is the bit-width of parameters' FXP representations, w_i and w_o are FLP weight and the quantized version, respectively, and a_i and a_o are FLP activation functions' output and the corresponding quantized version. w_o and a_o are

$$\begin{cases} w_o = 2\mathcal{Q}_k\left(\frac{\tanh(w_i)}{2\max(|\tanh(w_i)|)} + \frac{1}{2}\right) - 1 \\ a_o = 2\mathcal{Q}_k(a_i) \end{cases} \quad (2.7)$$

where $\mathcal{Q}_k(r)$ is defined as

$$\mathcal{Q}_k(r) = \frac{1}{2^k - 1} \text{round}((2^k - 1)r). \quad (2.8)$$

The combination of Equation 2.7 and 2.8 is a ‘straight-through estimator’ [36] that is also used in the back-propagation. And FLP parameters are quantized to the k -bit FXP format in the forward propagation. The quantization procedure was emulated in PyTorch, and it is a prototype for future hardware implementations. The performance of the proposed quantization strategy is evaluated in Figure 2.4, presented in Section 2.5.1.

2.4.2 Training Detail

As solving the MLE problem in Equation 2.4 is mathematically equivalent to minimizing the KL divergence [37], the KL divergence is adopted as the loss function to minimize the KL distance between the ground truth (GT) histograms and the network’s output histograms over the spatial dimension:

$$\begin{aligned}\mathcal{L}(H, \hat{H}, \theta) &= \frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M \sum_{n \in N} D_{KL}(h^{(m_1, m_2)}[n] || \hat{h}_{\theta}^{(m_1, m_2)}[n]) \\ &= \frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M \sum_{n \in N} h^{(m_1, m_2)}[n] \log \frac{h^{(m_1, m_2)}[n]}{\hat{h}_{\theta}^{(m_1, m_2)}[n]}\end{aligned}\tag{2.9}$$

where H is the histogram tensor composed of the pixel-wise GT histograms (denoted as h) and \hat{H} is the network output containing the predicted pixel-wise histograms (denoted as \hat{h}). To enhance the efficiency of 2D spatial denoising, the TV regularization [38] can minimize the spatial variation. The whole loss function becomes

$$\mathcal{L}(H, \hat{H}) = \mathcal{L}_{KL}(H, \hat{H}) + \beta_h TV(\mathcal{S}(\hat{H})),\tag{2.10}$$

where β_h is a tunable hyper-parameter before training. We use the SGD to learn the parameters during the back-propagation, expressed as

$$J(\theta) = \nabla_{\theta} \mathcal{L}(H, \hat{H}, \theta). \quad (2.11)$$

As for preparing training data, the NYUv2 [39] and Middlebury [40], datasets were utilized as the training and testing datasets, respectively. The training dataset contains nine types of indoor scenes with 13k synthetic ToA tensors, and the validation dataset contains 1.3k. The network’s input is the data cube with the size of $512 \times 512 \times 1024$, and the SGD with the ADAM algorithm [41] in the Pytorch library was employed to execute the back-propagation. Loss curves of training and validation shown in Figure 2.4 were generated and fetched from the Tensorboard toolkit. To achieve better visualization, we apply a smooth factor of 0.8 to alleviate fluctuations. For each convolutional module except the last, batch normalization and ReLU operations are added after the convolution operations to alleviate the vanishing gradient. We used a mini-batch to optimize the gradient descent and save memory with a batch size of 5. The hyper-parameter β_h of the TV loss function is 10^5 . 4 epochs (each contains 3200 iterations) are configured to train the network, guaranteeing a converging loss. The training dataset is randomly shuffled before each training epoch to generalize our model. Our architecture’s training time is 17 hours, 7 hours shorter than the existing sensor-fusion networks. For the inference phase, we aim to retrieve depth images with a high spatial resolution of 688×552 . Due to limited GPU memory, the whole tensor (with the size $588 \times 552 \times 1024$) was divided into small ones for the network’s input with the size $64 \times 64 \times 1024$. Since the loss curve of validation fluctuates significantly, it is difficult to use the early-stop method to

cease the training to prevent over-fitting. Our approach is to smooth the loss curve and obtain an approximate range that contains the smallest loss. Then we apply the synthetic testing dataset on these saved models, and we pick the one generating the minimum loss or RMSE. The selected model is employed to deduce the depth images from captured SPAD data. Finally, the reconstructed result with a high spatial resolution can be generated by seaming the individual low-resolution ones consecutively. We used SPAD and intensity data captured by the LinoSPAD system [42], as real-word test datasets. The tensor’s size is $256 \times 256 \times 1536$, and the average bin-width of embedded TDCs is 26 ps. Moreover, for a fair comparison with the monocular-SPAD architecture [30], DenseDepth [43] was used to reproduce monocular depth images to be fused with ToA data in the SPADnet [30].

2.5 Evaluation

2.5.1 Loss Evaluation

The training and validation loss of the FLP and the quantified model are shown in Figure 2.4. We implemented three quantization cases and compared them with the FLP version. Since the training and inference are susceptible to the activation’s bit-width, we tried different bit-widths to quantize the activations’ output and selected the best case. As depicted in Figure 2.4, only Case *W2A2* shows degraded performances. Therefore, considering the accuracy and consumption of computing resources, *W2A4* works the best. Table 2.1 compares the model sizes of existing sensor-fusion models and the proposed architecture, where ‘Lindell w/ intensity’ and ‘Lindell w/o intensity’ indicate the training with and without intensity images. The compression rate is determined by comparing the model

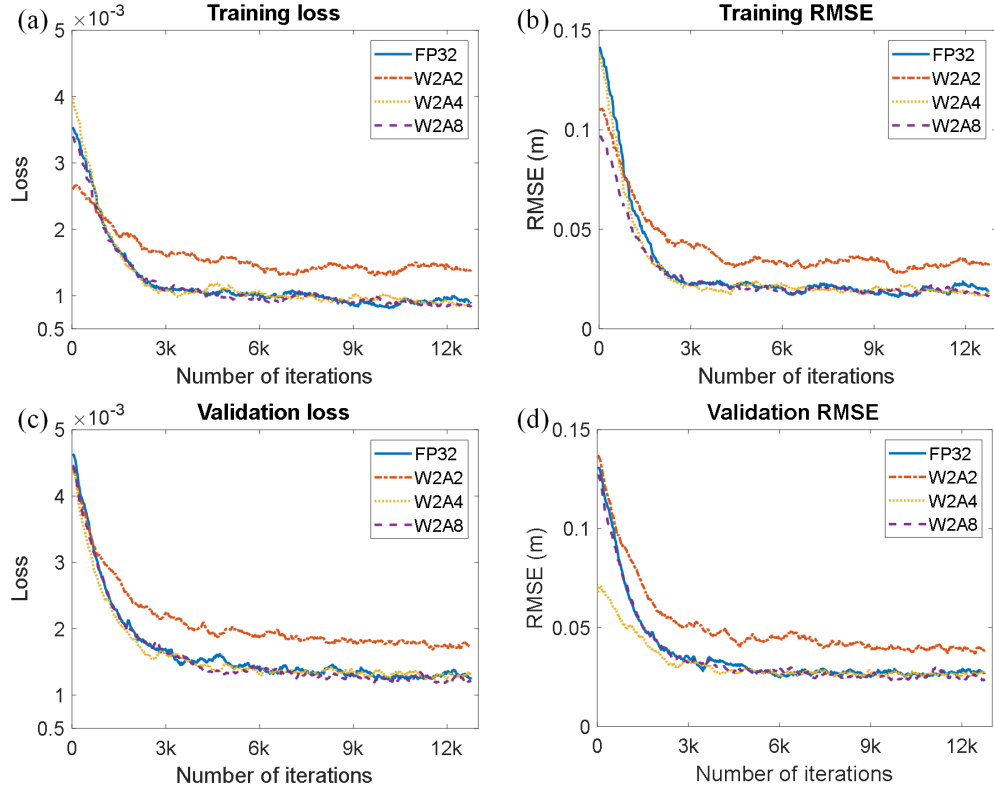


Figure 2.4: (a) and (b). Training and validation loss; (c) and (d). RMSE plots with different quantization cases. Each plot contains the original FLP 32-bit format and three quantization cases.

Table 2.1: Compression rates of our FLP model and existing networks in terms of the parameter size, training time and compression rate.

	Parameter size	Training time	Compression rate
Lindell w/ intensity	3.95 MB	24 h	$21.99 \times$
Lindell w/o intensity	3.93 MB	24 h	$21.93 \times$
Sun <i>et al.</i>	3.95 MB	24 h	$21.99 \times$
Non-local	1.01 MB	36 h	$5.61 \times$
Proposed (FLP 32-bit)	2.19 MB	17 h	$12.17 \times$
Proposed (W^2A_4)	0.18 MB	16 h	-

size (obtained from PyTorch) with existing models. Although our network was trained by less powerful GPUs (NVIDIA RTX 5000, whereas NVIDIA Titan V and 1080 Ti were used in [27], [30], [31]), it can still achieve the shortest training time. Moreover, after the quantization process, the network obtains a remarkable compression ratio compared with the original FLP model and existing network architectures. It is suitable for hardware-embedded solutions, as it requires much less on-memory to pre-load the model. The performance of the compressed model will be detailed in the following subsection.

2.5.2 Synthetic Data

We first evaluated the reconstruction quality for simulated data using five different metrics to assess the depth reconstruction. They are the accuracy of a given threshold thr , RMSE, RMSE (log), the absolute relative difference (*Abs rel*), and

the squared relative difference (*Sq rel*), defined to be

$$\begin{aligned}
RMSE(\mathbf{d}, \hat{\mathbf{d}}) &= \sqrt{\frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M (d_{m_1, m_2} - \hat{d}_{m_1, m_2})^2} \\
RMSE_{\log}(\mathbf{d}, \hat{\mathbf{d}}) &= \sqrt{\frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M (\log_{10}(d_{m_1, m_2}) - \log_{10}(\hat{d}_{m_1, m_2}))^2} \\
Abs\ rel(\mathbf{d}, \hat{\mathbf{d}}) &= \frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M \frac{|d_{m_1, m_2} - \hat{d}_{m_1, m_2}|}{d_{m_1, m_2}} \\
Sq\ rel(\mathbf{d}, \hat{\mathbf{d}}) &= \frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M \frac{(d_{m_1, m_2} - \hat{d}_{m_1, m_2})^2}{d_{m_1, m_2}}
\end{aligned} \tag{2.12}$$

and accuracy with threshold *thr*:

$$\text{percentage of } \hat{\mathbf{d}} \text{ s.t. } \frac{1}{M^2} \sum_{m_1=1}^M \sum_{m_2=1}^M \max\left(\frac{d_{m_1, m_2}}{\hat{d}_{m_1, m_2}}, \frac{\hat{d}_{m_1, m_2}}{d_{m_1, m_2}}\right) = \delta < thr, \tag{2.13}$$

where \mathbf{d} and $\hat{\mathbf{d}}$ denote the ground truth and predicted depth images.

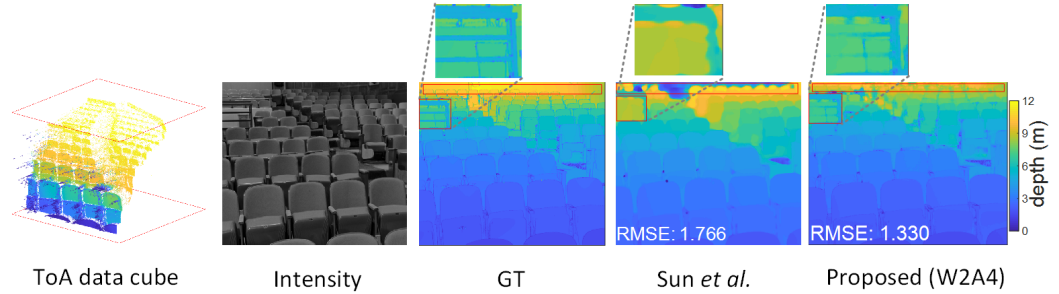


Figure 2.5: Depth images reconstructed from the ToA tensor. The fidelity differences between monocular-SPAD fusion and the proposed model are marked in red boxes. The proposed model can reveal more details for long-distance objects and obtain a lower overall RMSE when SBR equals 0.04.

Seven indoor target scenes of the Middlebury dataset were assessed. Table 2.3, shows the averaged values across seven scenes with three SBRs. Compared with existing algorithms, our compressed architecture obtains comparable or improved accuracy to Sun *et al.* [30] using RGB-SPAD fusion strategy and Peng *et al.* [31] using long- and short-range residual connections architecture. Despite the similar performances, we should notice that Sun *et al.* adopted a log-scale binning method merging fewer time-bins for the front indexes and more time-bins for the ending indexes; thereby, the accuracy deteriorates when sensing long-distance objects. We use a simulated indoor scene (a lecture theatre) to prove this point (shown in Figure 2.5) that the binning method cannot reconstruct relatively long objects highlighted in red boxes.

Table 2.2 shows our results are better than [30]. As for Peng *et al.*'s architecture, long-range correlations of feature maps were effectively explored, and dilated convolutions were employed to enlarge the reception field. Both methods can enhance the accuracy and generate fewer parameters. However, Peng *et al.*'s method has many vectorized operations during the training and inference phases, resulting in a longer training time (36 hours) than ours (17 hours with a less powerful GPU). Moreover, owing to the complicated residual connections of [31], it is difficult to reconfigure the structure for different tasks with a proper trade-off. Instead, our architecture has good scalability where the number of down- and up-sampling can be configured without redesigning the connections. Lastly, our network maintains high reconstruction accuracy despite a much smaller model size. To better indicate the statistical significance, we further calculated the p -value of our compressed model and the models in [30] and [31] in test datasets in terms of *Abs rel*. We obtained a list of *Abs rel* (in total 21 elements, representing 7 scenes in 3 SBR levels) from each pair of GT and reconstructed depth images

Table 2.2: Extensive evaluations between Sun *et al.*'s [23] and the proposed models with $\text{SBR} = 0.04$. The proposed model achieves more robust results across five evaluations metrics.

	Signal photons: 2; Background photons: 50; SBR: 0.04						
	Accuracy (Higher is better)			Error (Lower is better)			
	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	RMSE(log)	RMSE(m)	Abs rel	Sq rel
Sun <i>et al.</i>	0.9087	0.9503	0.9625	0.292	1.766	0.071	0.269
Proposed	0.8545	0.9618	0.9927	0.249	0.330	0.065	0.246

from each algorithm. The p -value of *Abs rel* is 2.94×10^{-5} between the model in [30] and ours (and 0.0162 between the model in [31] and ours). Both are smaller than 0.05, meaning *Abs rel* of our model is statistically significant versus [30] and [31].

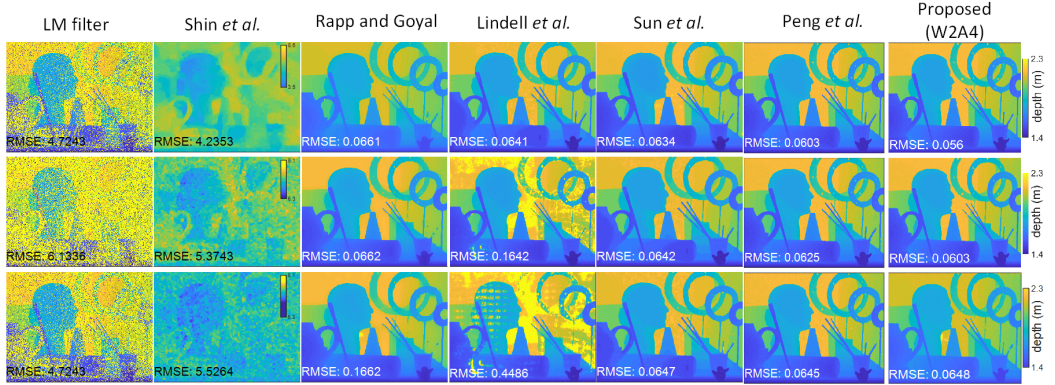


Figure 2.6: Reconstructed images using different algorithms. The SBRs from the first row to the last row are 0.2 (2 target photons and 10 background photons), 0.04 (2 target photons and 50 background photons), and 0.02 (2 target photons and 100 background photons). Our model can obtain lower RMSE than other algorithms when SBR equals 0.2 and 0.04 and achieves a comparable RMSE with Sun *et al.*, even without fusion strategies.

Table 2.3: Quantitative analysis of the proposed and existing algorithms over seven indoor scenes. Three tables are for three different SBR levels - 0.2, 0.04, and 0.02. And the underlined numbers mean that they are comparable to the best existing results.

Signal photons: 2; Background photons: 10; SBR: 0.2							
	Accuracy (Higher is better)			Error (Lower is better)			
	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	RMSE(log)	RMSE(m)	Abs rel	Sq rel
LM filter	0.5018	0.6115	0.6415	1.183	4.690	1.403	11.606
Shin <i>et al.</i>	0	0	0.0052	1.180	4.306	2.259	9.566
Rapp and Goyal	0.8691	0.9982	0.9998	1.102	0.063	0.0778	0.0310
Lindell <i>et al.</i>	0.9962	0.9982	<u>0.9999</u>	0.031	0.066	0.011	0.020
Peng <i>et al.</i>	0.9966	0.9981	<u>0.9999</u>	0.029	0.062	0.007	0.0015
Sun <i>et al.</i>	0.9966	0.9987	<u>0.9999</u>	0.030	0.064	0.0087	0.0019
Proposed (FLP)	0.9968	0.9983	<u>0.9999</u>	0.027	0.059	0.0069	0.0013
Proposed (<i>W2A4</i>)	0.9967	0.9980	<u>0.9999</u>	0.028	0.061	0.0056	0.0013
Signal photons: 2; Background photons: 50; SBR: 0.04							
LM filter	0.3492	0.4317	0.4737	1.184	5.774	2.070	17.615
Shin <i>et al.</i>	0	0	0	1.362	5.401	2.916	15.671
Rapp and Goyal <i>et al.</i>	0.8614	0.9976	0.9995	0.106	0.236	0.0780	0.0334
Lindell <i>et al.</i>	0.9827	0.9951	0.9999	0.064	0.149	0.026	0.0120
Peng <i>et al.</i>	0.9948	0.9971	0.9998	0.034	0.073	0.0082	0.0026
Sun <i>et al.</i>	<u>0.9961</u>	<u>0.9980</u>	<u>0.9999</u>	<u>0.030</u>	<u>0.064</u>	0.0087	0.0019
Proposed (FLP)	<u>0.9961</u>	<u>0.9980</u>	<u>0.9999</u>	0.029	0.063	0.0067	0.0017
Proposed (<i>W2A4</i>)	0.9962	<u>0.9980</u>	<u>0.9999</u>	<u>0.030</u>	<u>0.064</u>	0.0060	0.0016
Signal photons: 2; Background photons: 100; SBR: 0.02							
LM filter	0.2691	0.3439	0.3924	1.270	6.210	2.283	20.382
Shin <i>et al.</i>	0	0	0	1.383	5.620	2.990	16.568
Rapp and Goyal	0.8610	0.9974	0.9994	0.111	0.301	0.0790	0.0414
Lindell <i>et al.</i>	0.9357	0.9729	0.9902	0.129	0.321	0.0580	0.0600
Peng <i>et al.</i>	0.9952	0.9978	<u>0.9999</u>	0.033	0.069	0.0081	0.0021
Sun <i>et al.</i>	0.9961	<u>0.9981</u>	0.9999	0.031	<u>0.064</u>	0.0087	0.0020
Proposed (FLP)	0.9963	0.9980	<u>0.9999</u>	0.030	0.064	0.0060	0.0017
Proposed (<i>W2A4</i>)	0.9963	<u>0.9981</u>	<u>0.9999</u>	0.030	<u>0.065</u>	0.0060	0.0017

Since the compressed version ($W2A4$) can obtain almost the same performance as the FLP version, we employed it to conduct further visual comparisons hereafter. As shown in Figure 2.6, we chose one exemplar image named Art as an example. Rapp and Goyal’s algorithm achieves better recovering results than the LM filter and Shin *et al.*’s approach because they used pixel-wise signal refining and spatial regularization to smooth objects’ boundaries. However, their method is not robust in distinguishing object boundaries, whereas the neural networks in the last five columns (see Figure 2.6) can identify the boundaries through learning numerous training scenes. The proposed compressed model can achieve the smallest RMSE among existing methods.

2.6 Captured Data

We also tested the proposed compressed model ($W2A4$) and existing networks on five real-world image datasets. These scenes were captured in a low light condition (less than 1 photon per pixel) with low SBRs. In the first row of Figure 2.7, the reconstructed depth results from the LM filter [44] and Shin *et al.* [7] show the shadow that did not receive active illumination. In contrast, the other four algorithms can in-paint the shadow and generate relatively detailed depth maps. Our model can retrieve a more precise structure for the lamp scene. Although Rapp and Goyal adopted the super-pixel method to obtain robust depth estimations, some essential signals are lost, like the lamp’s circle part.

Similarly, in the second and the third rows, a rolling ball and an elephant toy are presented, as highlighted in red boxes. The human’s thumb and the ivory with distinct boundaries can be identified in our model’s depth images. However, these two small objects are over-smoothed by the monocular-SPAD fusion owing

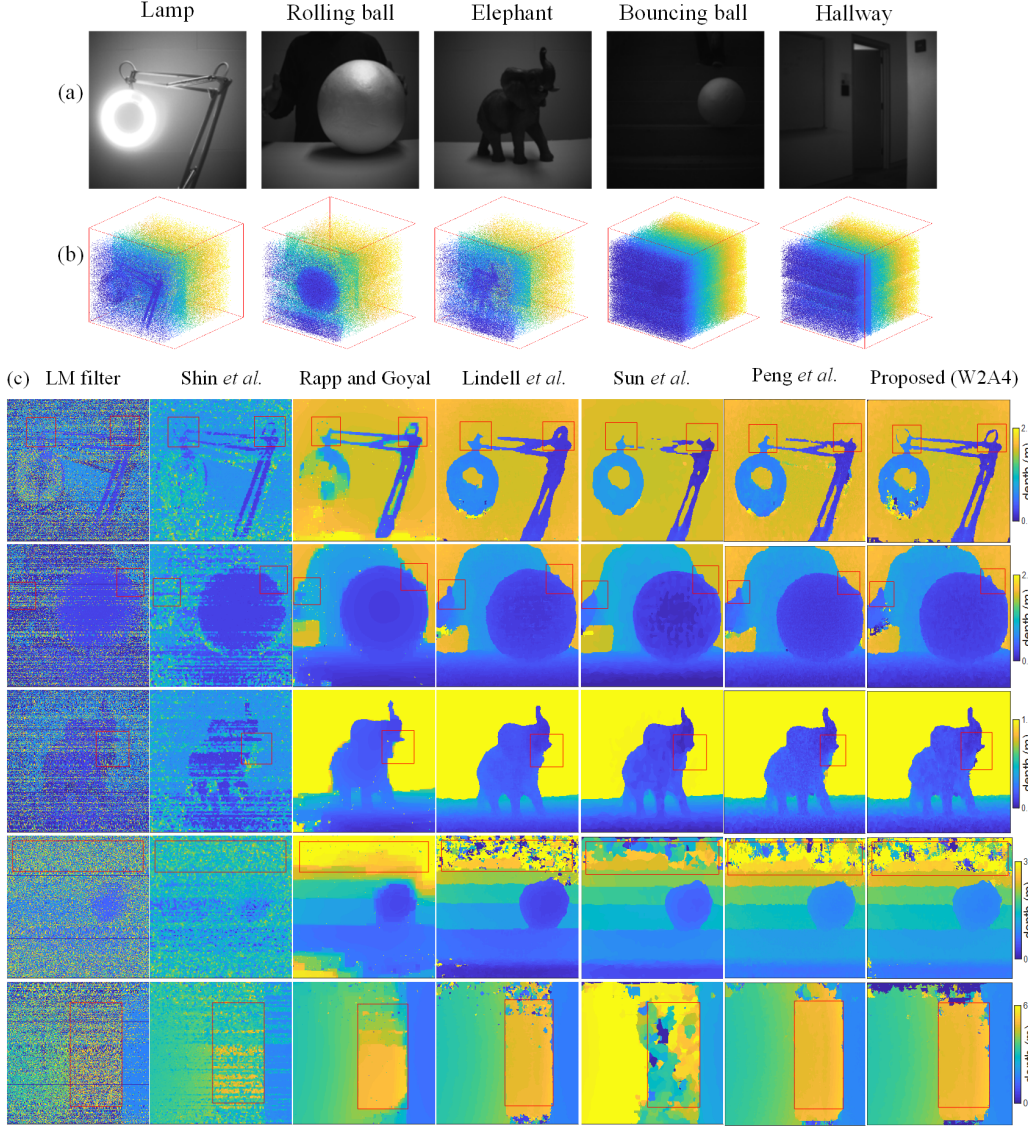


Figure 2.7: (a). Intensity images of the five scenes used for fusing with Lindell *et al.*'s and Sun *et al.*'s models. (b). Detected ToA data cubes of five scenes. The bouncing ball and a hallway (the 4th and 5th rows) scenes contain intense ambient light, leading to blurred impact. (c). Reconstructed depth maps. Lindell *et al.* used the intensity-SPAD fusion, and Sun *et al.* used the monocular-SPAD fusion. The processing time is proportional to the network's depth, and the RMSE decreases for a deeper network.

to the monocular perception. In the fourth row, the depth images of a bouncing ball from stairs are compared. The intense sunlight illumination at the top part of the scene leads to tremendous background noise. Rapp and Goyal’s spatially averaging strategy achieves relatively robust results. And for the last row, Peng *et al.*’s strategy can in-paint the spots corrupted by the ambient light; this might be due to the dense residual connections that can extract more features.

2.7 Discussion

We evaluated the proposed model with three depth levels, meaning the input tensor is downsampled three times. The fewer levels we use, the shorter processing time it costs (with fewer parameters). Figure 2.9 shows the network’s depth versus the processing speed and RMSE. L1 is a naive U-net architecture with one down-sampling module without long-range connections. As our architecture becomes deeper with both long- and short-range connections, the reconstruction error decreases significantly. The RMSEs for L4 to L2 are acceptable when the model’s depth reduces since they are better than existing studies. However, the processing time increases as more layers are added. Notably, due to the structural regularity of our model, it is easy to configure the numbers of down- and up-sampling without modifying the connections in the architecture. The log-scale binning method introduced by Sun *et al.* [30] can significantly reduce the GPU memory consumption and reduce the processing time. The binning method, however, would lead to critical degradation of performance. The configurability of our network can also be deemed the model pruning since fewer nodes are involved in computations when reducing the number of down-sampling. Therefore, our network is more robust to achieve a reasonable trade-off between processing time

and accuracy. The processing time is measured by inferencing one depth map from a ToA tensor with the size $552 \times 668 \times 1024$ under $\text{SBR}=0.04$. RMSE is an average value of seven scenes from the Middlebury dataset that are the same data used in Table 2.3.

It should be noted that there is no acceleration for our quantized architecture because the GPU cannot handle FXP operations due to the unified intrinsic hardware and instruction set architecture. However, FPGA can manipulate quantized numbers and accelerate the analysis; the hardware implementation is regarded as future work. Additionally, our architecture contains dense connections and multiple 3D convolution modules that consume the most processing time. ToA tensors are sparse with many zero elements involved in computing. Thus, more aggressive compression approaches like model pruning, sparse convolution strategies can help further accelerate the forward-propagation process.

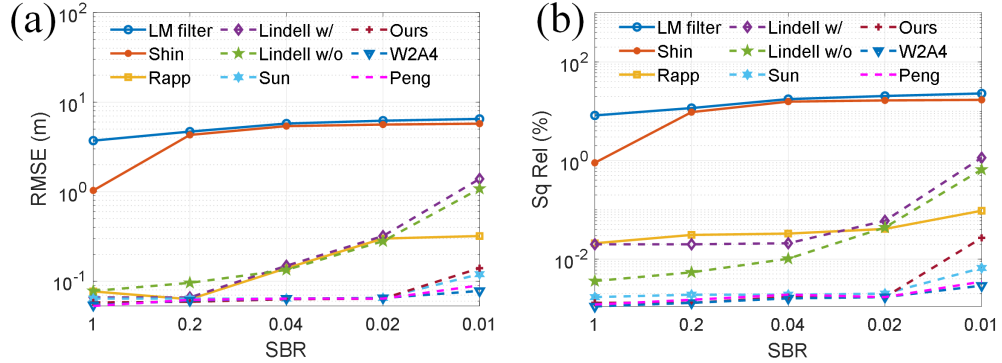


Figure 2.8: (a) RMSE and (b) squared relative difference (Sq rel) plots in terms of SBR. The dashed lines indicate neural network-based algorithms, and the rest represent optimization based algorithms. ‘Sun’ and ‘Lindell w/’ shown in the legend use the monocular-SPAD and intensity-SPAD fusion strategy, respectively.

The proposed network under different SBR conditions was further investigated in Figure 2.8. The algorithms using neural networks are more sensitive to background noise, especially when the SBR decreases from 0.02 to 0.01. We can increase the training dataset to cover a broader range of SBRs to alleviate this problem, although it costs more processing time. Nevertheless, the other three methods (LM filter, Shin *et al.*, and Rapp and Goyal) are more robust across all the SBRs. The LM filter and Shin *et al.*'s approach have maximum errors across all SBR levels. Rapp and Goyal's and Lindell *et al.*'s fusion models achieve relatively lower errors, and the former is more robust across all SBR levels. Sun *et al.*'s and the proposed network obtain the lowest errors and are robust for relatively low SBRs except the level 0.01. Therefore, it is feasible for the proposed neural network to achieve low prediction errors at standard SBR levels, and remain robust without using complementary training images from another camera. And optimization-based algorithms can be more stable than neural network-based methods over various SBR levels.

Despite relative better reconstruction results, our method still has limitations. For the scenes corrupted with enormous background noise, the time-bin index retrieved by the soft-argmax might be not the correct ToF profile. Whereas Rapp and Goyal [1] separated the signal and noise, a pixel-wise noise censoring algorithm and a windowing approach were introduced to remove noise and obtain the ToF information accurately. The reconstructed bouncing ball in Figure 2.7 shows that Rapp and Goyal's algorithm can produce relatively high fidelity in a noisy scenario.

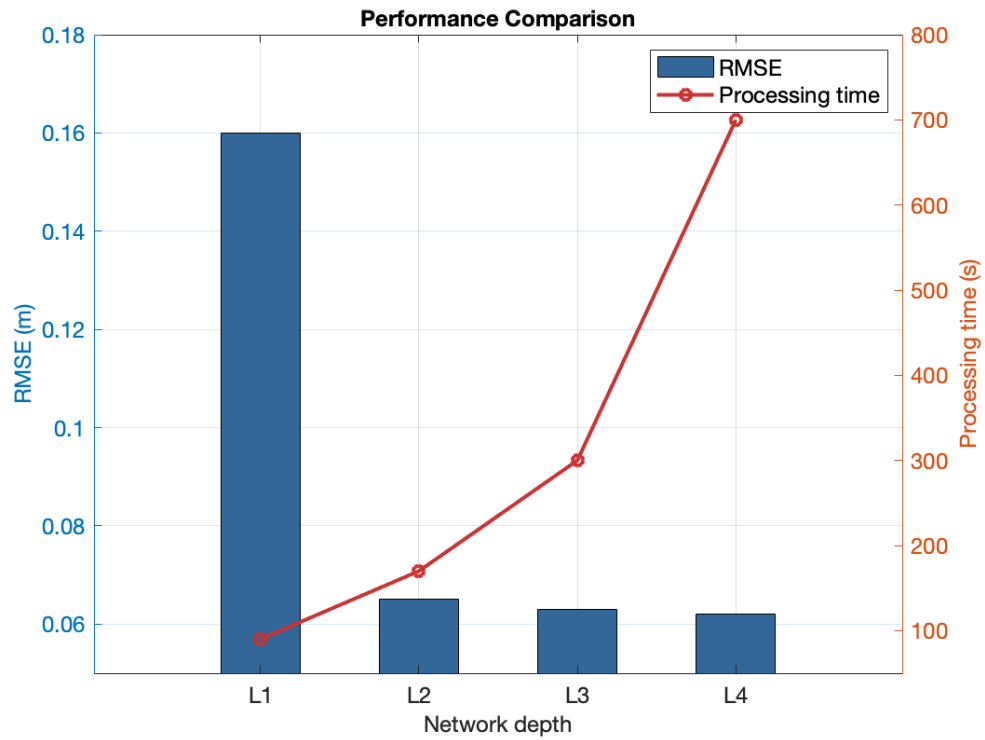


Figure 2.9: The processing time is proportional to the network's depth, and the RMSE decreases for a deeper network.

2.8 Summary and Future Work

We developed a non-fusion network that does not require an additional camera for depth estimation. It is much more practical for real-world applications. The proposed model was quantized to achieve a smaller model size without degrading performance. Due to the model’s flexible structure, the trade-off between accuracy and processing speed can be balanced by manipulating the network’s depth. It achieves excellent reconstruction performance in low light conditions and poor signal-to-background ratio conditions, and it can be extended to low photon-count biomedical computational imaging using single-photon avalanche diodes SPADs, such as time-domain FLIM. Accurate FLIM analysis in photon-starved conditions is challenging due to poor quantum yield, low photosensitivity, and suboptimal wavelength of excitation. The low photon emission deteriorates lifetime reconstruction. As reconstruction of fluorescence lifetime in FLIM and depth information in single-photon LiDAR both involve resolving ill-posed problems from histograms, the efficient deep neural network in this chapter is transferable to low-count FLIM. Another future work involves implementing the quantized network on portable hardware such as FPGAs and NVIDIA Jetson Nano GPUs. This effort aims to facilitate practical embedded single-photon LiDAR applications, such as long-range sensing and robotic vision.

Chapter 3

Compact and Robust Deep Learning Architecture for Fluorescence Lifetime Imaging and FPGA Implementation

3.1 Background

Fluorescence is represented by detectable photon emission from molecules or atoms, a radiative process transferring energy from an excited state to a lower state. Fluorescence lifetime is a quantitative metric to analyze fluorescence phenomena. FLIM is an emerging technique boosting biomedical imaging. For example, endogenous fluorophores, such as amino acids, metabolic enzymes, and vitamins, have been engineered to assess in *Vivo* imaging [45, 46]; exogenous fluorophores, such as FRET pairs, have been adopted to investigate protein interactions [47] and drug developments [48]. Autofluorescence has been utilized in fluorescence-guided surgery [49, 50]. TCSPC [51] is a powerful technique for time-resolved, wide-field [52], and confocal [53] FLIM as it exhibits high SNR of limited photons emitted from fluorophores [54]. To guarantee precise time-of-arrival, a

stopwatch employing high temporal resolution (picosecond) time-to-analogue or -digital converters can accumulate arrival photons from a train of laser pulses and generate a histogram decay. Retrieving lifetimes from TPSF (or histogram decays of collected photons) is an ill-posed problem in which a perfect solution cannot be found. Traditional deconvolution-based non-linear square fitting (NLSF) [55] and MLE [56] methods are computationally expensive.

Fast, model-free methods such as centre-of-mass (CMM) [3] and its derivatives [57] were proposed to estimate lifetimes without iterations to overcome the mentioned issues. However, they are susceptible to low SNRs. They can resolve either mono- or bi-exponential decays, meaning it is hard to analyze complex fluorophores with multi-exponential decays. Moreover, they cannot simultaneously and accurately recover two critical average lifetimes, i.e., amplitude- (τ_A) and intensity-weighted average lifetime (τ_I). The selection of the two average lifetimes should be considered for different applications [58]. This study clarified that τ_A is a crucial parameter to deduce energy transfer efficiency and dynamic quenching behaviours; τ_I is essential to solving the quenching constant from the Stern-Volmer constant. The study [59] clarified that the ratio of τ_A to τ_I is an intuitive indicator for monitoring multi-exponential fluorescence decays. Therefore, we aim to construct a model-free and robust method to reconstruct τ_A and τ_I to cover most FLIM applications. Given that generating artificial FLIM TPSFs is well-documented, synthetic data-powered data-driven approaches can boost the accuracy and computing efficiency of FLIM analysis. Therefore, we constructed a compact hardware-friendly DL architecture for high-accuracy τ_A and τ_I reconstruction. Further, we introduced a non-linear mapping approach to compress histograms without performance degradation.

On the other hand, an increasing number of computing platforms for biomedical devices are migrating toward the edges. For example, portable oximeters can measure oxygen concentration of flowing blood non-invasively using photoplethysmography. Such highly integrated devices perform efficient computing, yet the power budget is considerably low. Likely, we yearn to implement our DL networks on FPGA as it can achieve highly parallel, reconfigurable computing while consuming low power. We designed an automatic software script to extract, and export learned parameters from pre-trained models to FPGA, paving the way for other time-resolved biomedical applications. Further, we adopted on-chip post quantization to shorten the bit-width of the on-the-fly data. We analyzed the computing efficiency of the FPGA embedded with our neural networks and compared it with CPU and GPU.

3.2 Prior Work

This section summarizes existing work that utilizes DL to reconstruct fluorescence lifetime, and hardware accelerator to accelerate the reconstruction.

3.2.1 Deep Learning for FLIM

In the recent decade, DL has been an emerging approach for FLIM analysis. An MLP [60] was initially reported to analyze bi-exponential decays for a two-photon system. Another MLP model using skip-connections was proposed [61], which surpasses the conventional FLIM-MLP models in terms of accuracy. To build a noise-robust DL architecture, a 1D convolutional neural network (CNN) [62] was proposed to resolve bi- and tri-exponential decays with low SNRs. An MLP [4] was applied on a high spatial resolution, wide-field FLIM using time-gated SPAD

without raster-scanning to miniaturize the FLIM system. Another MLP [63] processes phasor coordinates and generates lifetime parameters. Because the inputs are coordinates rather than histograms, the MLP exhibits fast speed. To extract both spatial and temporal features from raw fluorescence data tensors, a 3D CNN [64] was reported for near infrared FLIM applications. Besides, Ruoyang *et al* proposed a 3D CNN combined with CS [65] for widefield FLIM imaging in vitro and in vivo environments. Further, to address the challenge of FLIM with photon-starved conditions, Yuan-I *et al.* reported a generative adversarial network [66] for FLIM imaging in low-photon scenarios (below 400). DL was also adopted to unmix fluorescence coefficients from spectrally overlapped fluorophores. Jason *et al.* proposed a hybrid CNN architecture [67] to extract individual lifetime components from multi-exponential, hyperspectral fluorescence emission decays. To enhance the spatial resolution of FLIM images, Dong *et al.* [68] introduced a cascade CNN architecture to infer fluorescence lifetime and improve the spatial resolution afterwards. An ELM [69] was presented to achieve fast and accurate lifetimes reconstruction with back-propagation-free, online training. Dong *et al.* [70] proposed a photon-efficient DL to reconstruct fluorescence lifetime with a few photons per pixel, using spatial correlation and intensity information.

3.2.2 Hardware Platforms for FLIM

GPUs have been extensively adopted as hardware accelerators for FLIM. Byungyeon *et al.* [71] proposed a mean-delay technique and implemented it on a GPU to directly extract fluorescence lifetime from analogue fluorescent decays. Additionally, a GPU was utilized to realize a pixel-wise, fitting-free phasor method [72] towards a two-photon FLIM system to achieve video-rate fluorescence lifetime inference. A compressed sensing algorithm was applied to a high-resolution, widefield

FLIM system that used compressed ultrafast photography [73]. A GPU-based computer cluster was used to solve the iterative reconstruction algorithm. An FPGA is the ideal hardware for data readout and computing due to its sufficient I/O interfaces and reconfigurable computing parallelism. Particularly, SPAD arrays [74] coupled with FPGA as parallel and high-throughput data readout are needed. Therefore, some FPGA friendly FLIM algorithms were proposed. Embedded CMM algorithms were implemented on FPGAs [75] and ASICs [76, 77] to determine lifetimes from TDCs' time stamps directly. Another CMM processor was implemented on an FPGA coupled with a SPAD-based silicon photomultiplier sensor [78], realizing real-time fluorescence lifetime estimation and online fluorescence lifetime cytometric sorting. After that, a DL prototype that is less susceptible to noise [79] was embedded on an FPGA to estimate fluorescence lifetime in a fluorescent flow cytometry system. On-chip lifetime estimation was also realized in the frequency domain; discrete Fourier transform was embedded on an FPGA to compute phase lifetimes and modulations [80]. An embedded recurrent neural network was realized on FPGA to reconstruct fluorescence lifetime [81], integrated into a SPAD-based FLIM microscope.

3.3 Problem Definition

Our initial objective is to retrieve lifetime parameters from TPSFs, where TPSF $h(t)$ for each focal point can be formulated by the convolution of the periodic IRF $I(t)$ and the multi-exponential PDF $p(t)$

$$\begin{aligned} h(t) &= I(t) * p(t) + \epsilon(t) \\ &= I(t) * \left[A \sum_{i=1}^N \alpha_i \exp\left(-\frac{t}{\tau_i}\right) \right] + \epsilon(t) \end{aligned} \tag{3.1}$$

where $\sum_{i=1}^N \alpha_i = 1$, A depicts the amplitude, α_i and τ_i mean the amplitude fractions and lifetime of an individual lifetime component, N implies the number of lifetime components. As Poisson noise is inevitable in TCSPC systems, $\epsilon(t)$ represents Poisson noise which is the square root of the total photons in each time bin in a decay curve. The intensity-weighted average lifetime τ_I means the average lifetime a fluorophore stays excited, depicted by

$$\tau_I = \frac{\int_0^\infty t \sum \alpha_i \exp\left(-\frac{t}{\tau_i}\right) dt}{\int_0^\infty \sum \alpha_i \exp\left(-\frac{t}{\tau_i}\right) dt} = \frac{\sum_{i=1}^N \alpha_i \tau_i^2}{\sum_{i=1}^N \alpha_i \tau_i} \quad (3.2)$$

An amplitude-weighted average lifetime is the integral of an intensity decay that is equivalent to steady-state intensity, formulated by

$$\tau_A = \int_0^\infty \sum \alpha_i \exp\left(-\frac{t}{\tau_i}\right) dt = \sum_{i=1}^N \alpha_i \tau_i \quad (3.3)$$

3.4 Deep Learning Network Details

We intend to achieve high-fidelity FLIM images with a minor computing overhead. The extreme case of lightweight DL networks is the binarized-neural network [82] that only involves bit-wise operations. However, it suffers from low accuracy in practical applications [83]. It was proved that 2D AdderNet [84] could achieve comparable accuracy with CNN without matrix multiplications. Here, we introduce a derivative of the original AdderNet, compressed 1D fluorescence lifetime AdderNet (FLAN), catering to the histogram decays' processing.

Algorithm 1 Unified AC for each histogram

Input: input FM F_i of shape $CH_i \times W_i$;
 filters W of shape $K_x \times CH_i \times CH_o$;
 BN coefficients *scale* and *shift* of shape CH_o ;
 stride S ; Threshold T ; photon count N_{pc} ;
Output: output FM F_o of shape $CH_o \times W_o$;

```

1: if  $N_{pc} > T$  then
2:   for  $ch_i = 0; ch_i < CH_i; ch_i ++$  do
3:     for  $w_o = 0; w_o < W_o; w_o ++$  do
4:       for  $k_x = 0; k_x < K_x; k_x ++$  do
5:          $w = ch_i \times S + k_x$ ;
6:         if  $(w \geq 0) \&\& (w \leq W_i)$  then
7:           while  $ch_o < CH_o$  do
8:              $F_o[w_o][ch_o] += -|F_i[w][ch_i]$ 
               $-W[k_x][ch_i][ch_o]|$ ;
9:              $ch_o ++$ ;
10:         $F_o = F_o \times scale[ch_o] + shift[ch_o]$ ;
11:         $F_o = ReLU(F_o)$ ;
12: else
13:    $F_o = 0$ ;
```

3.4.1 Network Architecture

The naïve AdderNet was initially used for 2D image classification [84]. It has been proved that adder-convolution (AC) using l_1 distance to calculate similarities between feature maps (FM) and kernels can achieve nearly identical accuracy versus traditional convolutions. We use strides in each AC instead of zero-padding to simplify hardware logic to enlarge reception fields. The schematic of FLAN is shown in Figure 3.1. We constructed UAC modules including AC, ReLU and batch-normalization (BN) to facilitate hardware implementation. And a ResNet [85] block was placed in the network’s backbone to alleviate gradient vanishing. Differentiating from our previous work [62] that extracting individual lifetime

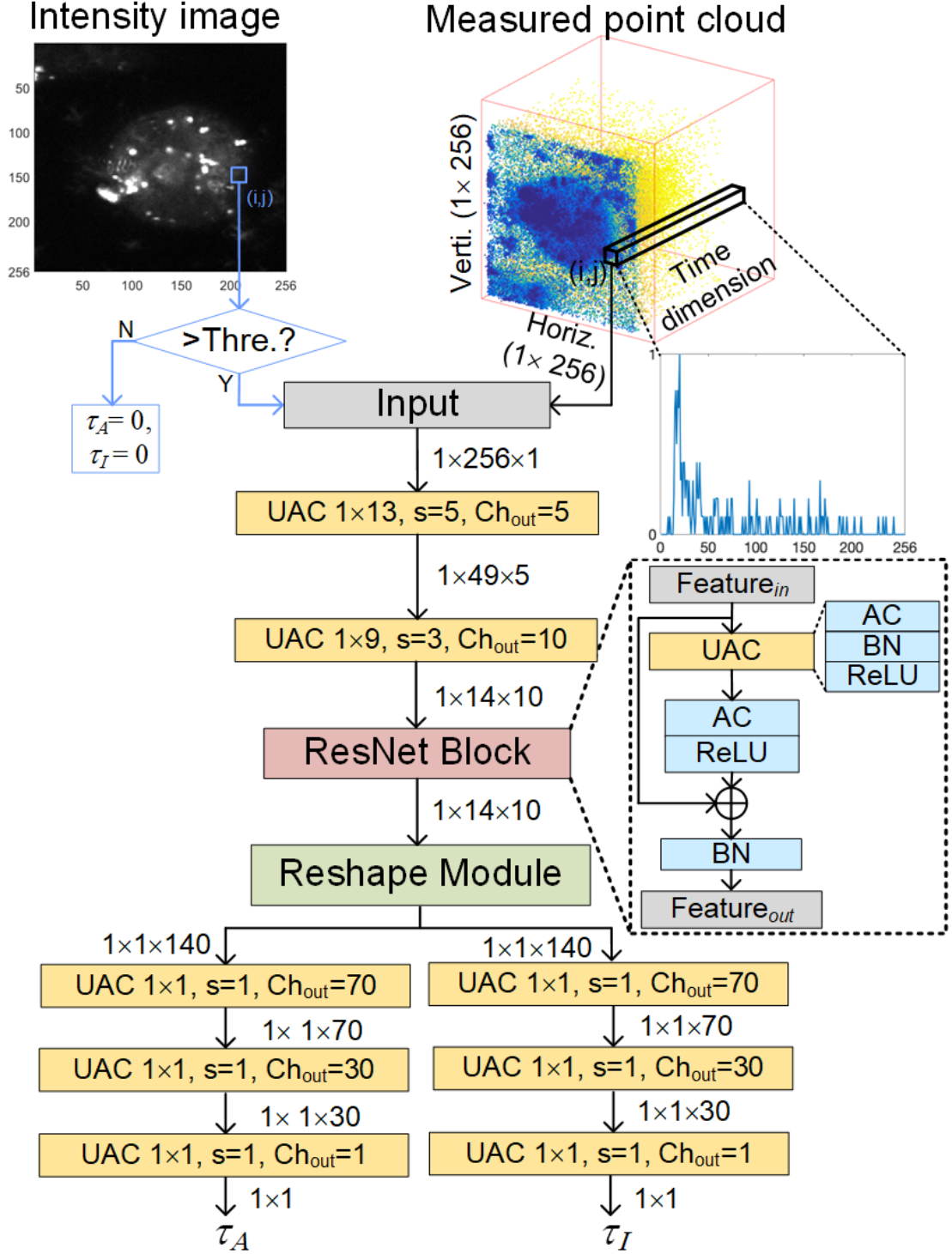


Figure 3.1: Overview of FLAN architecture. Histograms from each pixel will be processed consecutively, and the pixels from the background will be discarded initially. Prostatic cells coated with biomarkers are adopted here for illustration.

components, we adopted a branched topology to infer two average lifetimes τ_A and τ_I that can analyze most FLIM applications. Algorithm 1 demonstrates the entry and the first layer of FLAN. A pre-defined threshold (T) was applied to filter out background pixels (Line 1). If the pixel is not from the background ($N_{pc} > T$), a unified AC (UAC) will process the histogram (from Line 2 to 11). Moreover, we proposed FLAN + LS that compresses histograms using log-scale (LS) non-linear mapping. FLAN + LS needs fewer down-sampling layers because the length of input histograms was compressed in advance. Therefore, FLAN + LS is more compact and saves less computing overhead than FLAN. The details of compression will be illustrated in Section 3.6.

We compared the size of existing DLs for FLIM, and the compression ratio is defined by the ratio of FLAN + LS's parameter size to uncompressed models. Table 3.1 summarizes current DL architectures for FLIM. Among the existing architectures that can resolve multi-exponential decays, our models obtain the smallest parameter size apart from Phasor-MLP. Phasor-MLP has the fewest parameters as histograms were initially converted into phasor coordinates using the Fourier transform. Despite the simplicity, reconstructed lifetime parameters sometimes had distinct dispersion than 3D CNN [64]. ELM [69] has the simplest structure containing only one hidden layer, where the hidden layer contains 500 nodes. ELM's computational complexity exponentially increases as the number of nodes grows. Such a complex, fully connected structure is not efficient for hardware implementation. Besides, although the 1D CNN [79] for fluorescence flow cytometry has a compact size and was optimized and implemented on FPGA, it cannot analyze multi-exponential decays. As the data throughput of photomultiplier tube (PMT) or SPAD array is enormous (Gigabyte per second), lightweight DL archi-

Table 3.1: Comparisons of DL Architectures for FLIM

	3D CNN [64]	1D CNN [79]	1D CNN [62]	MLP [60]	Phasor-MLP [63]	MLP [4]	GAN [66]	ELM [69]	FLAN	FLAN+LS
Multi-exponential compatible	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Input data type	3D Tensor	1D histogram	1D histogram	1D histogram	Phasor coordinate	1D histogram	1D histogram	1D histogram	1D histogram	1D histogram
Model size	4.14 MB (1 084 045)	0.016 MB (4284)	0.19 MB (48 675)	0.57 MB (149 252)	1.84 KB (471)	14.31 MB (3 750 250)	0.55 MB (143 528 ^a)	0.13 MB (205 600)	0.088 MB (23 003)	0.017 MB (4058)
Training time	4 h	17 min	23 min	4 h	15 min	38 min	6.9 h	10.85 sec	20 min	18 min
Training hardware	GPU	GPU	GPU	GPU	CPU	GPU	GPU	CPU	GPU	GPU
Inference hardware	GPU	GPU or FPGA	GPU	GPU	CPU	GPU	GPU	CPU	GPU or FPGA	GPU or FPGA
Data compression	-	Quantization-aware training	-	-	-	-	-	-	Post-quantization	Post-quantization + LS

^aThe model size was not given in the paper; the size was calculated from the network structure.

texture and efficient data compression are imperative to alleviate the bandwidth of the data transfer. FLAN and FLAN + LS can efficiently address the issues with the ACs and histogram compression method.

3.4.2 Preparation for Hybrid Training Data

As bi-exponential models could well approximate multi-exponential decays [59], we generated hybrid datasets containing both mono- and bi-exponential decays according to Equation 3.1 to cover most cases of fluorescence decays. The generation flow is shown in Figure 3.2. The synthetic IRF with 0.167 ns FWHM was simulated following a Gaussian curve centre at the 14th time bin that is consistent with our two-photon system, given by

$$I(t) = \exp\{-2[\frac{(t - 14)2(0.5 \ln 2)^{1/2N}}{FWHM}]^{2N}\}, \quad (3.4)$$

where we set $N = 1$ to make the synthetic IRF close to the real IRF. The mono- and bi-exponential probability dense functions (PDFs) were generated using the same instrumental parameters. Then bespoke TPSF were generated by convolving the IRF with PDFs. Poisson noise was added, ultimately using *Poisrnd* (·) in MATLAB[®] to mimic background and shot noise. Peak photon counts (N_p) and lifetime parameters (τ for mono-exponential, τ_I , τ_A , and α for bi-exponential decays) were randomly selected in given ranges, as shown in Table 3.2. Each synthetic TPSF was labelled with GT lifetime parameters to train our networks afterwards. Then, these two types of TPSFs were shuffled to finalize the datasets.

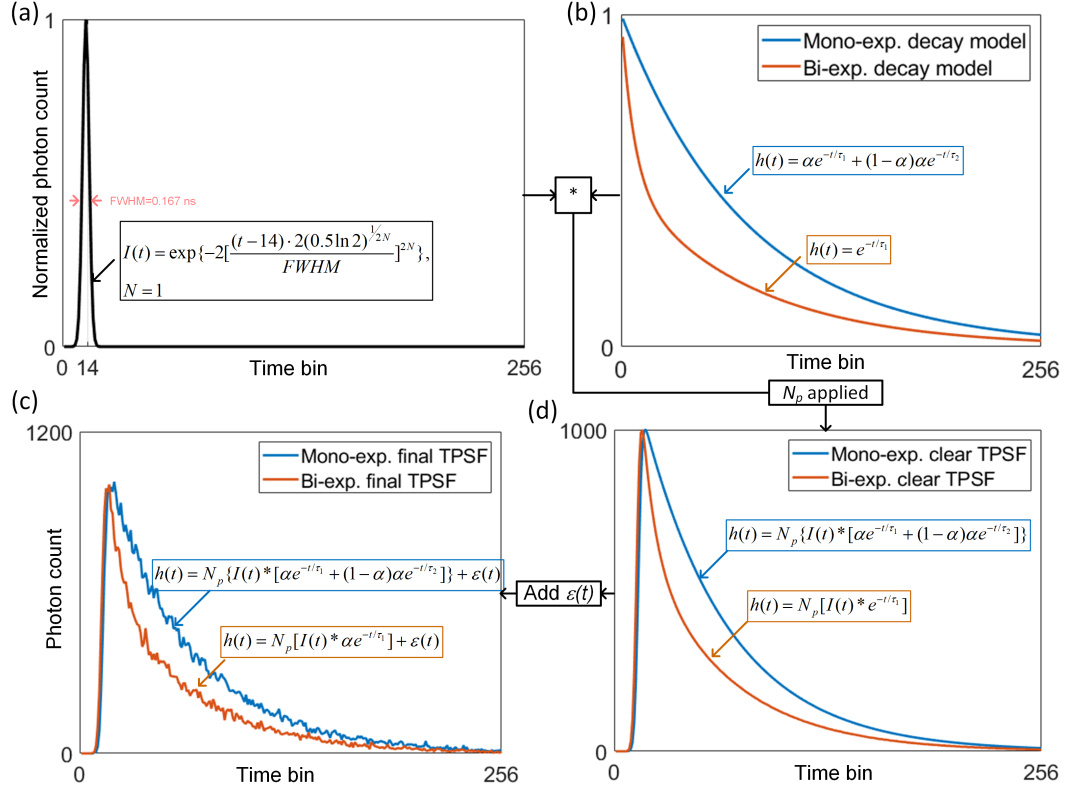
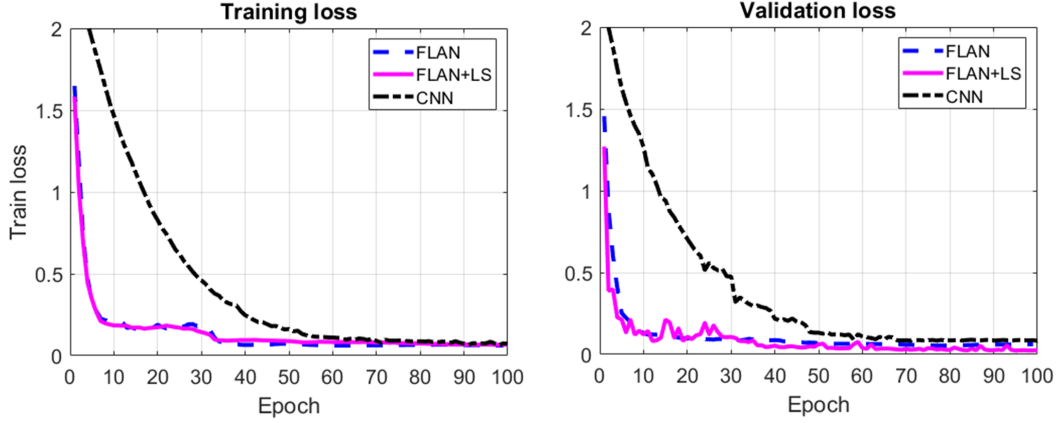


Figure 3.2: Procedures for generating mono- and bi-exponential TPSFs. The simulated IRF $I(t)$ convolved with mono- and bi-exponential PDFs $h(t)$. After that, peak photon counts N_p were applied to mimic real histograms. And then Poisson noise $\epsilon(t)$ was added at the final step.

Table 3.2: Ranges of lifetime parameters for synthetic data.

	$\tau_1(\text{ns})$	$\tau_2(\text{ns})$	α	$\tau_I(\text{ns})$	$\tau_A(\text{ns})$	N_p
Mono-exp.	[0.1, 5]	-	1	[0.1, 5]	[0.1, 5]	[10, 400]
Bi-exp.	[0.1, 5]	[1, 3]	[0, 1]	[0.01, 3.98]	[0.01, 3.98]	[10, 400]

**Figure 3.3:** Training and validation loss curves from 1D CNN, FLAN, and FLAN + LS.

3.5 Training Details

FLAN and FLAN + LS architectures were implemented by PyTorch and trained on a 16 GB NVIDIA Quadro RTX 5000 GPU. To accelerate the training, we employed adaptive learning rates (ALR) with an initial value of 0.001 and a momentum factor of 0.995. RMSProp is the optimizer in the training phase. 50,000 synthetic decays were generated for training purposes; extra 5,000 decays were utilized for validation. Early-stopping with 20-epoch patience was adopted to prevent overfitting. ReLU is the activation function after each AC layer. BN was used to mitigate the internal covariate shift of FMs in the network, whereby

the training can be accelerated. The adopted loss function is MSE, depicted as

$$\mathcal{L}(\theta) = \frac{1}{B_1} \sum_{b_1}^{B_1} \sum_{b_2}^{B_2} \|F_{b_2}(h^{b_1}, \theta) - G^{b_1}\|_2^2, \quad (3.5)$$

where B_1 and B_2 are the batch size (128) and the number of output nodes (2), $F(\cdot)$ is the end-to-end inference function of training parameters θ and G is the set of ground truth lifetimes. To make comparisons, we re-trained the 1D CNN [62] with traditional multiplications-based convolutional kernels for comparison, where we kept the networks' topology identical to FLAN and FLAN + LS. As the learning rate (0.0001) of the original 1D CNN is constant, it would exhibit a slower convergent speed than ALR. The loss curves of the three networks are shown in Figure 3.3. FLAN + LS achieves a similar loss curve versus FLAN even with fewer time bins. And these two networks obtained similar ending loss versus 1D CNN when training ceased.

3.6 Data Compression Strategies

We introduce two-fold data compression methods to alleviate computing latency and reduce data movements. At first, we compressed histograms using a logscale bin-merging technique to reduce redundant time bins. And then, a post-linear-quantization was employed on FPGA to compress bit-width throughout the FLAN and FLAN + LS architecture.

3.6.1 Log-Scale Mapping for Time-Bin

The fluorescence decays are less informative as the index of time-bin increases, especially for the specimen with ultra-small lifetimes.

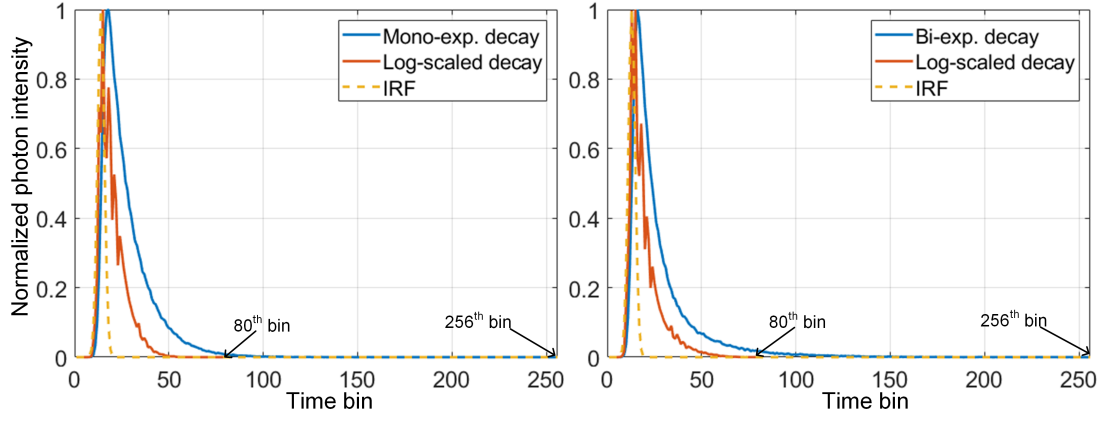


Figure 3.4: IRFs, original and compressed synthetic decays. (a) Mono-exponential decay with lifetime. (b) Bi-exponential decay with lifetime.

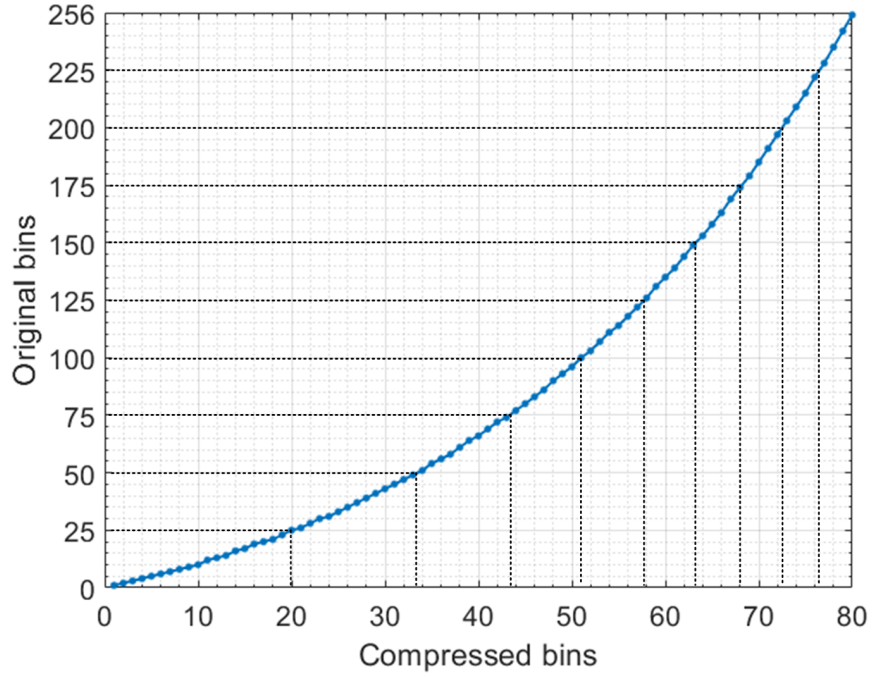


Figure 3.5: Nonlinear time-bin mapping, merging original 256 time-bin to compressed 80 time-bin.

As blue lines show in Figure 3.4, they are mono-and bi-exponential decays with lifetimes of 0.5 ns and 0.45 ns, respectively. No photon occurred after 86th and 84th time-bin, meaning that time bins behind them can be merged to compress the histogram and moderate the sparsity. Therefore, the computational complexity can be decreased. The target number of time bins M was controlled by a factor that should be predefined; the time-bin interval of the compressed histogram is

$$\begin{aligned} S &= s(x+1) - s(x) \\ &= \lfloor \frac{r^x - 1}{r - 1} \rfloor - \lfloor \frac{r^{x+1} - 1}{r - 1} \rfloor, \end{aligned} \tag{3.6}$$

where $\text{floor}(\cdot)$ rounds each element to the nearest integer less than or equal to that element, T is the original number of time-bin (256), x is the index of compressed time-bin. r is calculated from M , given by

$$\frac{r^M - 1}{r - 1} = T, \tag{3.7}$$

where M is 80 in our experiments because the compressed histograms can reserve sufficient temporal information after merging. The number of photons in the new time interval is the summation of the counts from the original intervals.

The non-linear mapping process is depicted in Figure 3.5; more temporal information can be reserved for the front bins. Accordingly, in Figure 3.4, both typically synthesized mono- and bi-exponential decays with 256 time-bin can be effectively converted into 80 bins. Compressed decays with fewer time-bin will simplify the neural network. For the decays with greater lifetimes, M can be easily adjusted; and the training data can be generated accordingly.

3.6.2 On-Chip Linear Quantization

Quantization for FMs and learned parameters are well-known approximate arithmetic for DL. And it is economical for logic utilization and power consumption on FPGA. We adopted the linear quantization (or static fixed-point) strategy [86] to convert FLP learned parameters and FMs in pre-trained models into FXP. To maintain the computing accuracy, we configured FMs with 16-bit integer and 16-bit fractional length segmentation; learned parameters with 10-bit integer and 10-bit fractional parts. The conversion can be depicted as

$$\begin{cases} D = \text{round}(N_{FLP} \times 2^F) \\ B = D2B(B) \\ N_{FXP} = \text{shift}(B, F) \end{cases} \quad (3.8)$$

Where F represents the fractional length of the FXP number, $D2B$ refers to converting from a decimal to a binary representation. For FM values, the $\text{shift}(\cdot)$ operation moves the decimal point 16 places to the left to convert the number to an FXP format. N_{FLP} and N_{FXP} denote the FLP number and the binarized FXP number, respectively. The precision of FXP numbers can be calculated using 2^{-N} , where N is the number of fractional bits. Quantization can be implemented using FXP libraries in Vivado HLS.

3.7 Synthetic Data Evaluation

With hybrid test datasets containing shuffled mono- and bi-exponential decays, we quantitatively evaluated FLAN and FLAN + LS in terms of τ_A and τ_I reconstruction. And we compared them with traditional curve fitting and other

high-accuracy algorithms in different photon-count conditions. To generate intuitive 2D GT FLIM images, two individual lifetime components (τ_1, τ_2 in Equation 3.1) were fixed values of 0.3 ns and 2.5 ns. And from the top to the bottom of GT lifetimes, amplitude fractions uniformly increased from 0 to 1.

MSE was used to evaluate the accuracy of each method in Figure 3.6. Three DL methods achieved better recovery performance for both τ_A and τ_I . And both FLAN and FLAN + LS outperformed 1D CNN under different photon-count conditions with the same training datasets. Despite the compression of FLAN + LS, the performance was not deteriorated, even in low-count conditions. In terms of τ_A , the performance of NLSF declined as photon counts decreased because the fitting process (Levenberg-Marquart deconvolution) required enough counts to guarantee accuracy. The previous study [87] using similar synthetic parameters to ours unraveled that, for mono-exponential decays, the necessary number of photons of the deconvolution method is around 100 for the minimal resolvable lifetime of 0.3 ns. Moreover, pre-set initial values are critical for the deconvolution method, and we did not fine-tune them here. BCMM was also susceptible to the low-count condition (below 100) as Romberg’s integration in BCMM is inaccurate for sparse vectors. As for τ_I , in Figures 3.6 (d) and (e), although CMM achieved smaller MSE than DL methods under high and medium photon count conditions, discernable bias can be observed in the top half of CMM’s recovered images.

Before using CMM, prior knowledge about the width of the analysis window is required to achieve optimal results. To observe the lifetime distributions more efficiently, we visualized the lifetime distributions of all recovered and GT images in Figure 3.7.

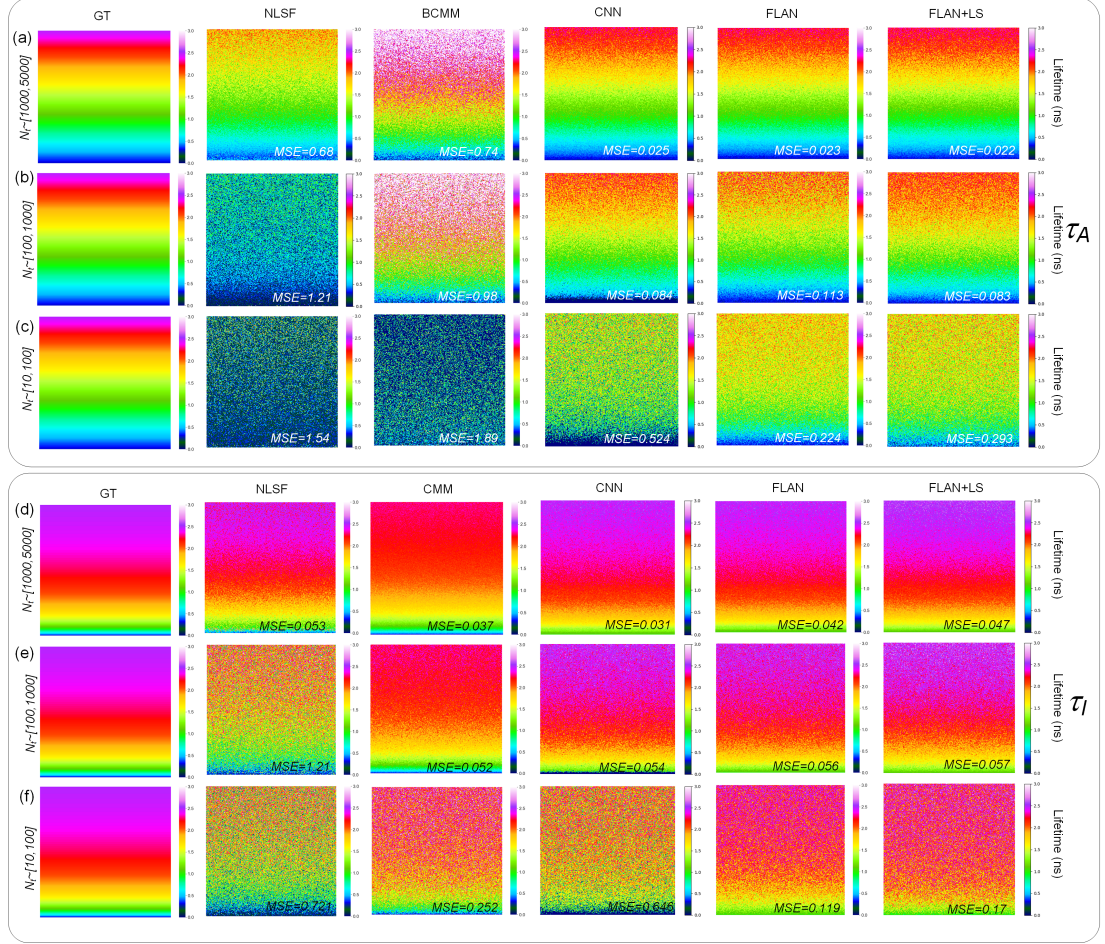


Figure 3.6: Synthetic GT and reconstructed τ_A and τ_I FLIM images. MSE is used to evaluate the accuracy. (a),(b), and (c) reconstructed τ_A in $[1000, 5000]$, $[100, 1000]$, and $[10, 100]$ photon-count. (d),(e), and (f) reconstructed τ_I in $[1000, 5000]$, $[100, 1000]$, and $[10, 100]$ photon-count.

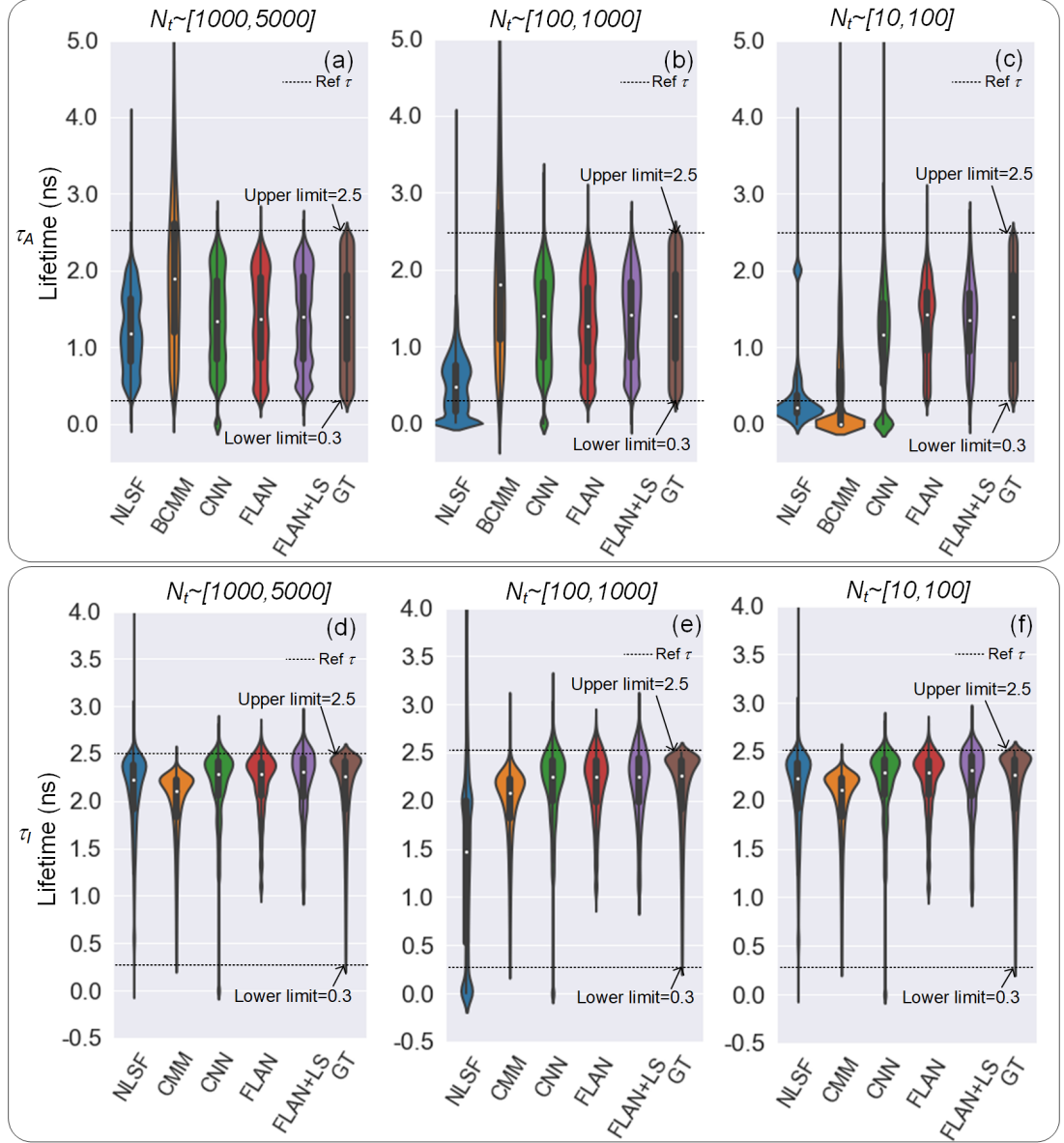


Figure 3.7: Lifetime distributions of τ_A in (a),(b), and (c); τ_I in (d),(e), and (f) retrieved from different algorithms in different levels of photon count.

3.8 Real-Case Study: Fluorescent Beads Discrimination

To evaluate our networks using fluorescent samples, we used different algorithms to retrieve lifetimes based on fluorescent beads with characterized lifetimes. Moreover, we used *convallaria majalis* cells with short acquisition time (3 seconds) to validate the accuracy of our algorithms in real low counts conditions. This section introduces the sample preparation processes, optical setup, and evaluation results.

3.8.1 Sample Preparation

We chose fluorescent latex beads to evaluate the accuracy of our networks as they are intuitive tools applied to tracing fluid dynamics [88] and validating hyperspectral imaging microscopy [89]. Here, yellowgreen and crimson fluorescent beads of 10 μm diameter (F8831, FluoSphere Polystyrene Microspheres, Thermo Fisher, UK) were dissolved in an aqueous concentration of $3.6 \cdot 10^5$ beads/ml. We dropped the diluted solution of fluorescent beads onto a microscope slide covered with coverslips and then mounted it on the microscope sample holder of a twophoton system. The unmixed set encompasses yellowgreen beads with a reference lifetime of 2.1 ns. Along with the unmixed group, the mixed set includes extra crimson beads with a 3 ns reference lifetime. The reference lifetime was measured using the TCSPC technique on a Horiba Deltaflex fluorometer and analyzed using Horiba DAS6 software.

3.8.2 Optical Setup

The utilized excitation source was a femtosecond Ti: sapphire laser (Chameleon, Coherent, Santa Clara, USA) with tunable wavelengths, and the laser pulse width is less than 200 fs with an 80 MHz repetition rate. The emission light was collected by a $10\times$ objective lens (N.A = 0.25) with a 685 nm short pass filter. Our setup acquired both sets of beads using LSM 510 confocal laser scanning microscope. A TCSPC (SPC-830, Becker & Hickl GmbH) card was mounted on the computer with a PCIe interface to record the time-of-flight of fluorescence emission. The spatial resolution of the FLIM data was configured as 256×256 on the SPImage software, and the field of view was selected by moving the translation stage. The number of time bins was configured as 256 in each pixel, and 39.06 ps was the temporal resolution in each time-bin.

3.8.3 Quantitative Analysis

Given that phasor [90] is a fitting-free, frequency domain approach to observing lifetime distributions represented by polar coordinates g and s

$$\begin{cases} g_{i,j} = \frac{\int_0^\infty h(t)_{i,j} \cos(kwt) dt}{\int_0^\infty h(t)_{i,j} dt} \\ s_{i,j} = \frac{\int_0^\infty h(t)_{i,j} \sin(kwt) dt}{\int_0^\infty h(t)_{i,j} dt} \end{cases}, \quad (3.9)$$

where i and j is the pixels' index in the FLIM image, kw is the angular frequency of the excitation source. Here, we initially used a phasor plot to analyze distributions of lifetimes in each pixel to validate the prepared beads. As shown in Figure 3.8, the cluster is not located in the semi-circle, meaning that the two types of beads are not perfect mono- and bi-exponential decays resulting from a

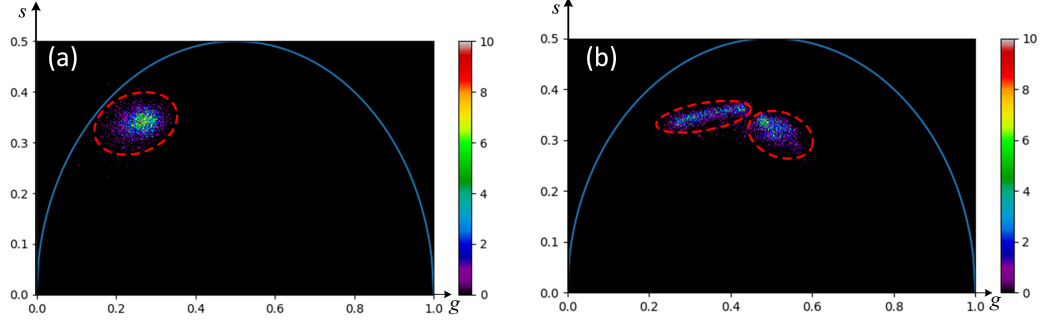


Figure 3.8: Phasor plots of unmixed (Yellow-green) fluorescent beads and mixed (Yellow-green and Crimson), showing (a) one cluster and (b) two clusters.

mixture of dyes. Given that there is no spectral overlap of the beads, meaning no energy transfer occurs, we used τ_I to analyze reconstructed lifetime images and to compare different algorithms. Two metrics, i.e. accuracy and precision in dB [79], were used for evaluation, defined as

$$\begin{cases} Acc. = 20 \log_{10} mean(\hat{\tau}_I / \Delta\tau_I) \\ Pre. = 20 \log_{10} mean(\hat{\tau}_I / \delta\tau_I) \end{cases}, \quad (3.10)$$

where $\hat{\tau}_I$ is reconstructed lifetimes, $\Delta\tau_I$ and $\delta\tau_I$ means absolute error and standard deviation of the reconstructed lifetimes. From Figure 3.9, we can tell DL methods obtained similar accuracy versus CMM, yet FLAN and FLAN + LS exhibit the best precision (small standard deviations). Note that NLSF has an obvious bias compared to others because it involves iterative deconvolutions that significantly rely on initial values. Likewise, we used τ_I to evaluate the algorithms for mixed beads. According to Figure 3.8 (b), as there were two individual clusters, we applied different algorithms to discriminate between these two types of beads. Reconstructed FLIM images and lifetime distributions from each algorithm were plotted in Figure 3.10 and 3.11. The CNN is less discerning than FLAN and

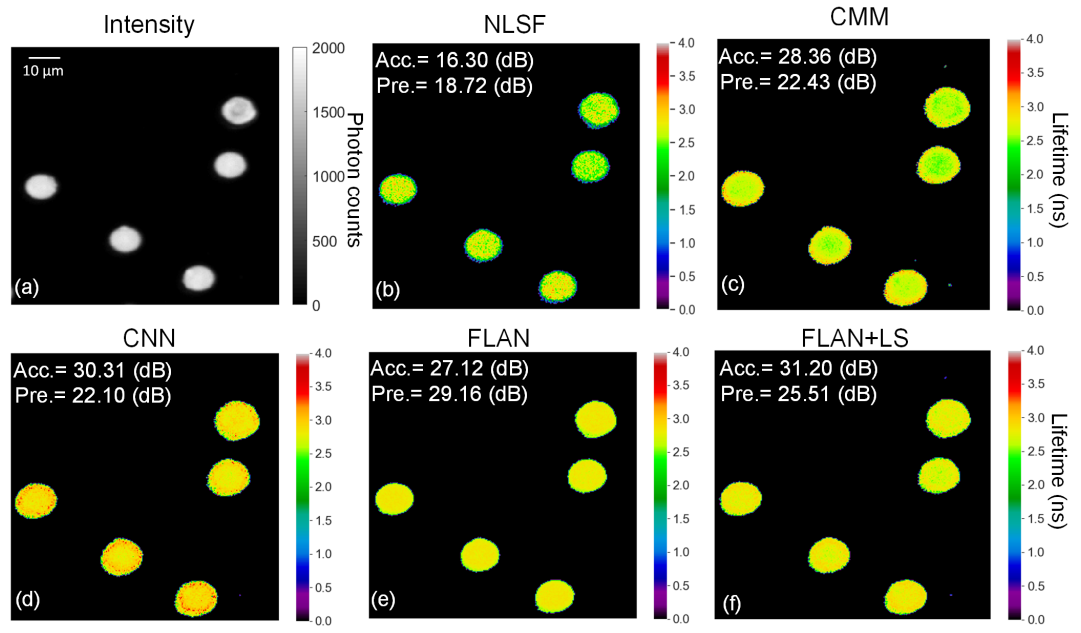


Figure 3.9: Unmixed beads (Yellow-green) evaluation using accuracy and precision

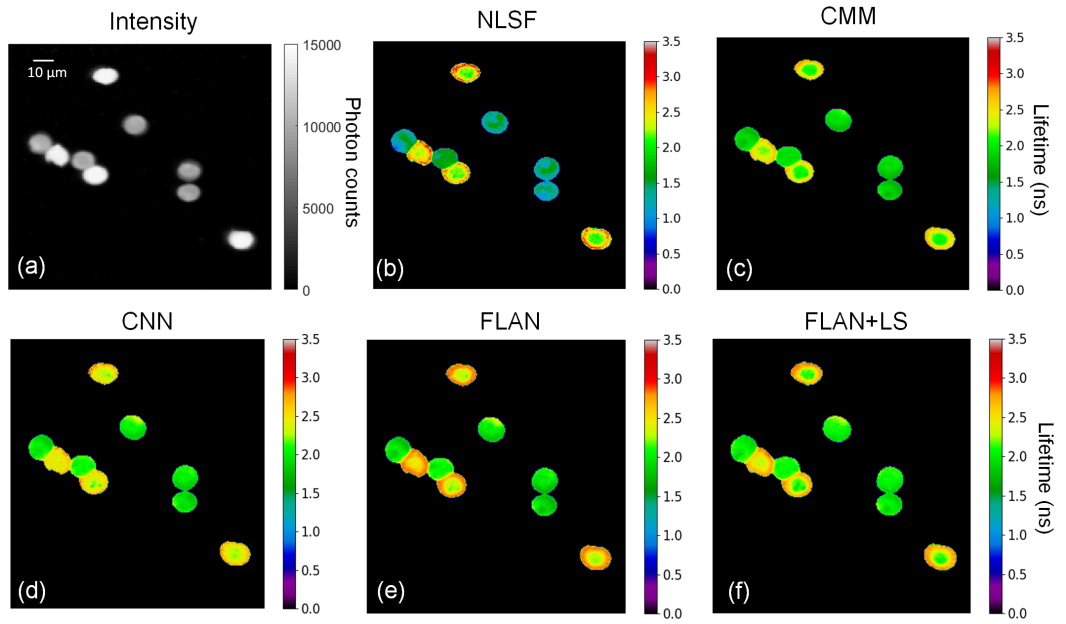


Figure 3.10: Lifetime reconstruction of mixed beads (Yellow-green and Crimson).

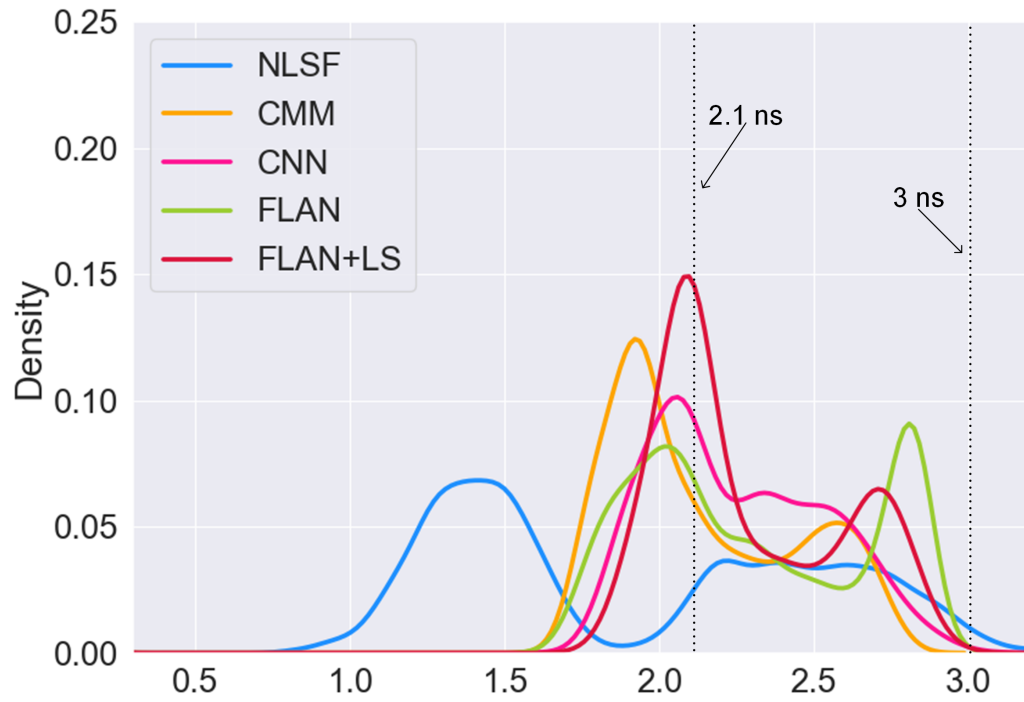


Figure 3.11: Reconstructed lifetime distributions of mixed beads using different algorithms. The reference lifetimes of Yellow-green and Crimson are 2.1 ns and 3 ns, respectively.

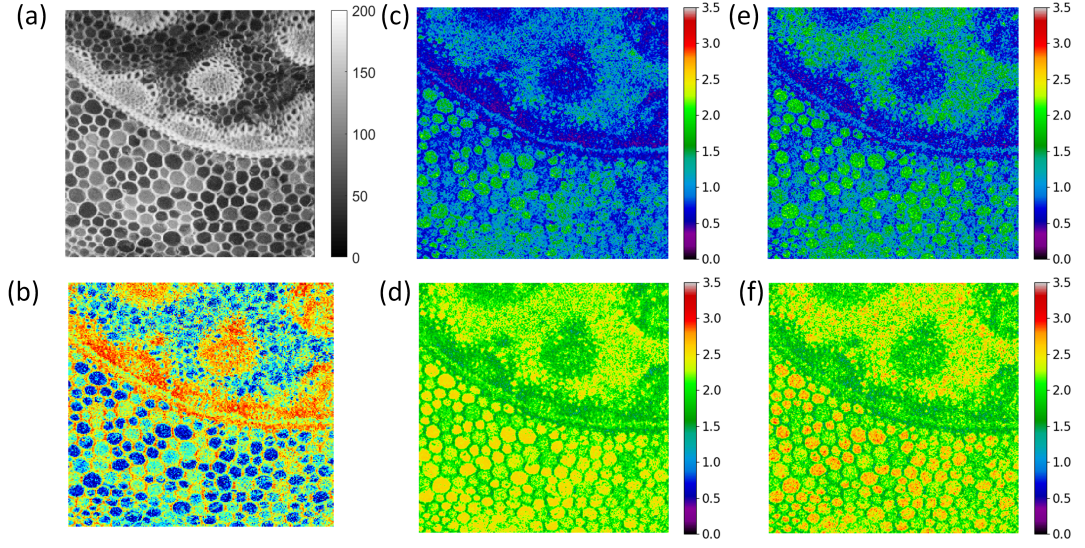


Figure 3.12: Reconstructed lifetime images of *Convallaria majalis* cells. (a) intensity image; (b) phasor projection image; (c) and (d) reconstructed τ_A and τ_I from FLAN; (e) and (f) reconstructed τ_A and τ_I from FLAN+LS.

FLAN + LS to discriminate between the two types of beads. CMM can distinguish the beads well, but bias still exists. Peaks' indexes of FLAN and FLAN + LS are closest to the reference lifetime. Hence, our networks resolve two types of beads with accuracy lifetime estimation.

Figure 3.12 shows reconstructed lifetime results for low counts condition where the maximum count is 200. Figure 3.12 (b) is the phasor projection image calculated from high counts decays (with maximum counts of 2,500) with a longer acquisition time (15 seconds). Phasor projection images were leveraged as a reference to investigate the contrast of lifetime images [62, 69]. Figure 3.12 (c) and (d) indicate reconstructed τ_A and τ_I from FLAN; Figure 3.12 (e) and (f) indicate reconstructed τ_A and τ_I from FLAN + LS. FLAN and FLAN + LS exhibit similar contrast compared to high-count phasor protection image, meaning that our algorithms are also robust in real low-count conditions.

3.9 Hardware Implementation

The FLAN and FLAN + LS algorithms were implemented on a PYNQ-Z2 board integrating the processing system (PS) and the programmable logic (PL). The DL cores were implemented in the PL part using Vivado[®] high-level synthesis (HLS). Hardware drivers of the cores and less computing-intensive arithmetic, such as eliminating background pixels and decoding input FLP histograms into FP, were programmed in the PS.

As shown in Figure 3.14, four DL cores were instantiated to process four pixels concurrently to achieve instances-level parallel computing. The four consecutive histograms from an AXI bus (through a high-performance slave port) should be copied into four memory buffers. And then, the four chunks of data will be fed into corresponding DL cores. Learned parameters, including weights and BN, were fetched from the pre-trained model and exported into BRAMs using an automatic Python script. Thanks to the quantization strategy, all the parameters encoded in shortened bit-width save memories and can be prestored in on-chip BRAMs instead of off-chip DDR memory, which decreases the latency of data movement. Before computing lifetimes, according to the predefined threshold, a binarized mask map with zeros and ones should be generated offline. Only the pixels' indexes with ones will be processed. Similar to learned parameters, original input histograms should also be encoded from FLP to FXP. This encoding process was conducted in the ARM processor. Eventually, the output lifetimes will be converted to readable FLP after lifetime's inference. As mentioned in Section 3.4.1, gathered AC, ReLU, and BN can simplify the hardware implementation; we merge the three modules into one hardware module, indicated by grey areas next to the main AC modules in Figure 3.15 (a). The original channel-wise BN

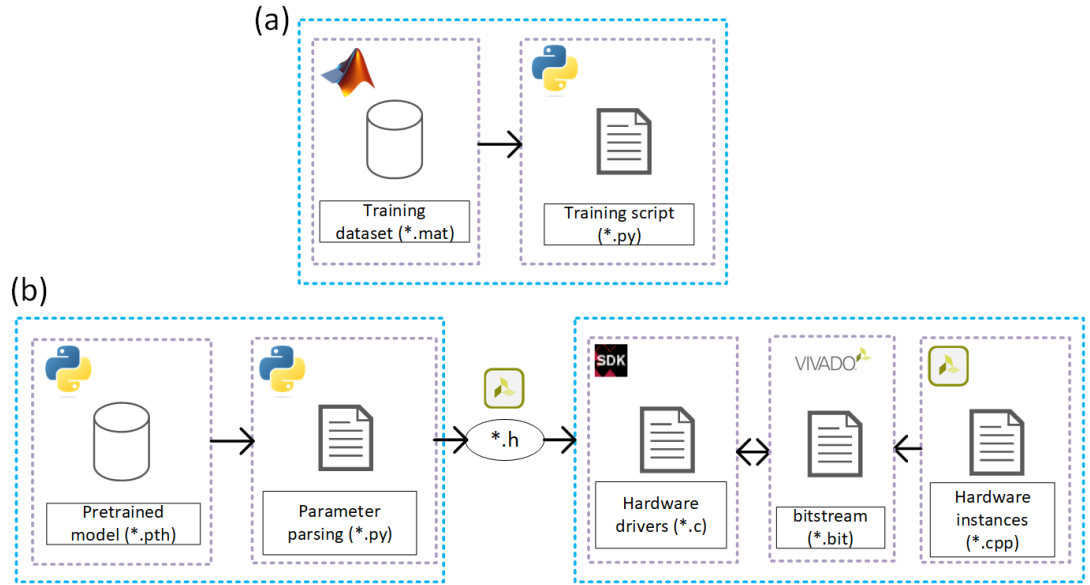


Figure 3.13: Pipelines of files generation for (a) training on PC and (b) inference on FPGA.

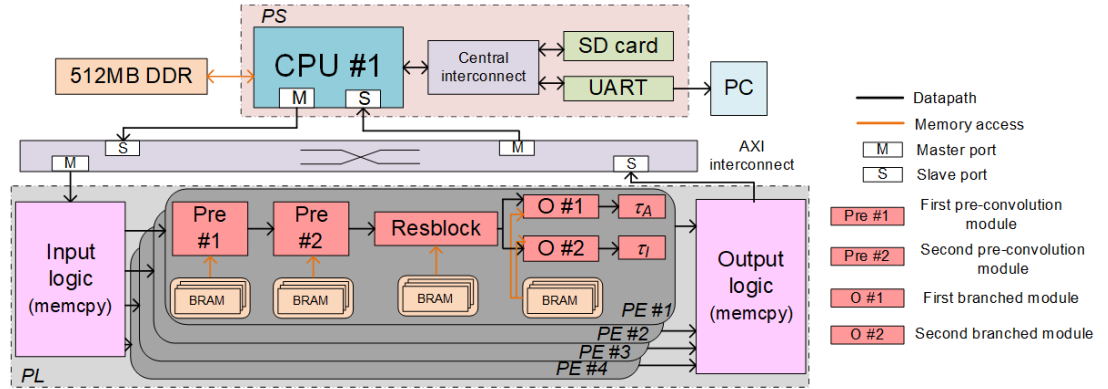


Figure 3.14: Overview of the FPGA hardware platform embedding four FLANs.

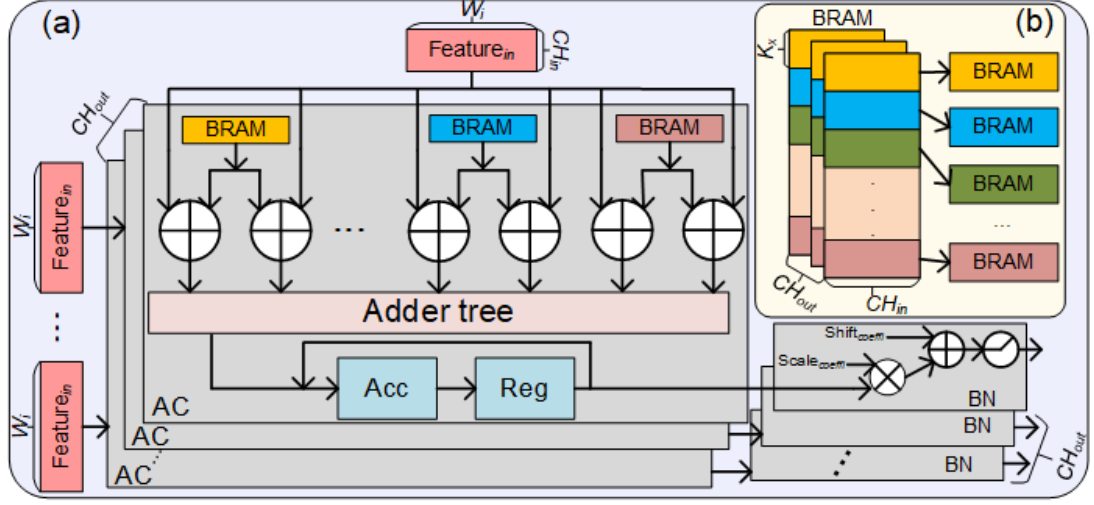


Figure 3.15: Architecture of UAC. BRAMs storing and caching learned parameters and FMs are partitioned to feed data to corresponding ACs. Multiple ACs and BN modules are instantiated to improve parallelism. (b) BRAMs were partitioned into smaller portions to cache parallel data on-the-fly.

operation in i_{th} output channel of forwards propagation is

$$x_{BN} = \frac{(x(i) - \mu(i))}{\sqrt{\delta(i)^2 + \epsilon}} \gamma(i) + \beta(i), \quad (3.11)$$

where γ and β are learned parameters that can be parsed from pre-trained models, μ_{BN} and δ_{BN}^2 are the statistical mean and standard deviation of x , ϵ is a constant to avoid dividing by zero (0.0001 by default). The scaling and shift coefficients can be extracted as

$$\begin{cases} scale(i) = \frac{\gamma(i)}{\sqrt{\delta_{BN}(i)^2 + \epsilon}} \\ shift(i) = \beta(i) - \frac{\gamma(i)\mu_{BN}(i)}{\sqrt{\delta_{BN}(i)^2 + \epsilon}}. \end{cases} \quad (3.12)$$

Therefore, BN can be simplified as matrix multiplications and vector additions on the hardware

$$x_{BN} = scale(i) \times x(i) + shift(i), \quad (3.13)$$

scale and *shift* were calculated and extracted offline using a Python script. As limited writing and reading ports of BRAM would hinder simultaneous data transfer to multiple PEs. To overcome the challenge, BRAMs under usage were partitioned into small portions to increase write-read bandwidth, leading to more hardware consumption. Due to our models' compaction, all the intermediate results among layers were cached in on-chip BRAM instead of accessing the DDR memory. After processing four adjacent pixels, only a vector containing eight inferred lifetime values will be output to ARM via address-mapping AXI for anti-quantization decoding and validation.

Figure 3.13 depicts the flow from training to inference. Notably, equations 3.12 and 3.13 are automated in Figure 3.13(b), making them transferable to new models. Additionally, hardware instances implemented using HLS were parameterized for flexible configuration of model size. Table 3.3 compares hardware utilization and computing performance of three pixel-wise DL networks, namely, 1D CNN with 32-b FLP, FLAN with 16-b FXP, and FLAN + LS with 16-b FXP, implemented on FPGA. The accuracy was assessed by MSE using synthetic test datasets. We randomly picked 100 histograms and divided them into 25 batches as our DL processors can process 4 pixels concurrently. And the batches of histograms were stored in the DDR. For 1D CNN, only 2 DL cores can be accommodated due to more hardware consumption of FLP computing, whereby the 100 histograms should be divided into 50 batches. The latency of the three networks is presented by the consumed time for processing 4 pixels.

Throughput was defined by pixels per millisecond (PMS). MSE was adopted to assess the accuracy, aligning with Figure 3.6. The hardware utilization was obtained after place-and-route implementation. The power consumption was estimated in the post-implementation phase using power analyzer tools in Vivado

Table 3.3: Comparisons of Hardware-Efficient DL Networks Implemented on FPGA.

	1-D CNN (2-core)	FLAN (4-core)	FLAN+LS (4-core)
Frequency	98 MHz	110 MHz	120 MHz
Precision	32 b FLP	16 b FXP	16 b FXP
DSP (220 Ava.)	51 (23.2 %)	100 (45.4 %)	76 (34.6 %)
LUT (53.2K Ava.)	51 072 (95.8 %)	25 886 (48.6 %)	18 553 (34.9 %)
DFF (106.3K Ava.)	30 827 (28.7 %)	18 816 (17.7 %)	15 629 (14.7 %)
BRAM (140 Ava.)	41 (29.1 %)	124.5 (88.9 %)	47 (33.6 %)
LUTRAM (17.4K Ava.)	1566 (9 %)	2202 (12.7 %)	2198 (12.6 %)
τ_A (MSE)	0.121 ns	0.125 ns	0.138 ns
τ_I (MSE)	0.089 ns	0.083 ns	0.102 ns
Power (W)	2.56	1.93	1.78
Latency (ms)	1.73	0.16	0.11
Throughout (PMS)	15	33	40
Efficiency (PMS/W)	5.86	17.10	22.47

2018.2. FLAN and FLAN + LS with shorter bit-width save multitudes of hardware and achieve high operating frequency. FLAN and FLAN + LS still utilize DSPs as multiplications of BN were synthesized using DSP. Although four DL units were implemented in FLAN + LS, it consumes the fewest LUT and DFF. Due to the numerous partitioned memory blocks in FLAN and FLAN + LS, more distributed (LUT) RAMs were consumed to meet the memory bandwidth. Despite the on-chip quantization, there was no apparent accuracy degradation for τ_A and τ_I reconstruction.

We also investigated how batch sizes would affect the performance of FLAN + LS on CPU, GPU, and FPGA, depicted in Figure 3.16. GPU’s computing performance significantly relied on the batch size and the scale of DL models. Small batch size and model size would dramatically deteriorate the speed. The figure shows that CPU and GPU present similar throughput when the batch size is smaller than 32. GPU’s throughput exceeds CPU when the batch size exceeds

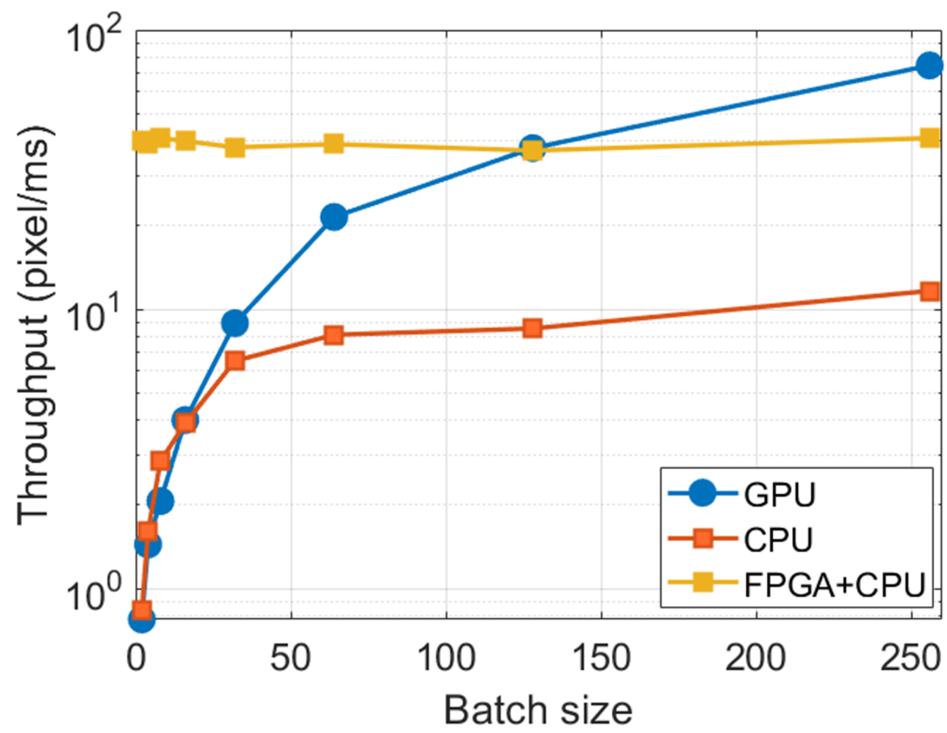


Figure 3.16: Relationships between batch sizes and throughput (pixel/ms) in the inference phase on CPU, GPU, and FPGA + CPU.

32 and FPGA when the batch size is more significant than 128. Note that, for some FLIM microscope applications like ours, the contrast between the sample and background should be distinguished, whereby each pixel should be compared to the threshold. This compels GPU process histograms pixel-by-pixel in serial (batch size equals 1) instead of parallel. Moreover, pixel-wise data movement on a GPU consumes extra time. Another reason hindering GPU’s performance is that CUDA and cuDNN are not providing optimizations for adder-based convolutions. As for FPGAs, they are not restricted by batch sizes, and they excel at processing data streams and addition operations.

3.10 Summary and Future Work

This study reports a multiplication-free and robust DL network for time-domain FLIM analysis. Besides, two data compression strategies were introduced to accelerate the processing. Both synthetic and real data were adopted to extensively and quantitatively demonstrate superior results comparing existing algorithms. Besides, to illustrate the applicability of hardware, we implemented our DL architectures on FPGA and compared them with traditional 1D CNN on hardware. Results show that our compression methods can significantly save hardware resources and achieve high computational efficiency without performance deterioration. Also, we investigate our lightweight DL network’s performance on GPU, CPU, and FPGA+CPU. We conclude that our heterogeneous FPGA+CPU is not affected by batch sizes and outperforms GPU for small batch sizes. A potential improvement of this work is that the training datasets and the network architecture are tailored for confocal microscopes that require a scanning mechanism and long acquisition time to guarantee high SNR of the captured data. Whereas for

wide-field FLIM, imaging systems using SPAD array do not use scanning strategies but widefield illumination, leading to shorter acquisition time and low SNR. And IRFs vary for different systems. Therefore, training datasets and the IRF should be modified to cater to various single-photon sensors. Further, our DL hardware cores cannot be integrated with the TCSPC module coupled with the confocal microscope due to the non-configurable property of the TCSPC board. However, it is possible to embed the DL cores in the data readout circuit of SPAD arrays because FPGA acts as the readout and control hardware for SPAD arrays. We will make our DL more generalized for SPAD array-based systems in software and hardware aspects.

Apart from FLIM, the DL network can be applied to analyze data from diffuse correlation spectroscopy (DCS) [91] that also relies on single-photon sensors and TCSPC systems. For DCS, the essential task is to retrieve scattering coefficients and blood flow index from temporal correlation functions. To migrate the workflow in this work to DCS, we need to alter the theoretical modelling of training datasets and the optical path, and this work paves the way for DCS research.

Chapter 4

Fast Fluorescence Lifetime Imaging Analysis Using Extreme Learning Machine for Time-Domain Single-Photon Detector

4.1 Background

As discussed in Section 3.1, 1D DL exhibits superior accuracy and computational efficiency while reconstructing lifetimes. However, DL’s accuracy may deteriorate when the input data comes from a different optical setup with varying instrument specifications, such as IRF, temporal resolution of TDC, quantum efficiency, etc. Although transfer learning can alleviate this effect, risks such as domain shift, overfitting, and computational overhead [92] still exist. Our goal is to address the generalization issue via an online learning algorithm. The extreme learning machine (ELM) [93] offers an efficient solution for processing 1D signals in biological applications, such as electrocardiogram (ECG) and electroen-

cephalogram (EEG) signals [94]. Inspired by related literature, we employed ELM to reconstruct lifetimes from 1D histograms using multi-variable regression. Here, the contributions of the ELM-based lifetime inference approach include:

1. The propagation-free property in ELM training was utilized to learn the datasets, achieving shorter training times compared to existing ANN methods. This paves the way for fast online training on embedded hardware for FLIM.
2. Mono- and bi-exponential models, widely used in practical experiments, were employed to evaluate the performance of ELM, focusing on the amplitude and intensity average lifetimes.
3. Reconstructed lifetime parameters from ELM are more accurate than those from fitting and non-fitting algorithms for both synthetic and experimental data under various photon-counting conditions, while maintaining fast computational speed.

4.2 Apply ELM to FLIM

Due to ELM's superior capability of processing 1-D signals, we associated synthetic 1D histograms with ELM regarding training and inferencing phases. We also illustrate the probabilistic model of photon arrivals of FLIM data and the artificial IRF based on TCSPC.

4.2.1 ELM Theory

Conventionally, back-propagation is the gold standard to minimize object functions in most ANN architectures. ELM is theoretically a single-layer feed-forward network (SLFN) that uses matrix inversion (or Moore–Penrose matrix inversion) and minimum norm least-square solution to train models. The training can be accelerated significantly compared with iterative back-propagation procedures whilst avoiding slow convergence and over-fitting resulting from back-propagation. Assume H training samples (H pairs of vectors $x_i = [x_{i1}, x_{i2}, \dots, x_{im}]^T \in \mathbb{R}^m$ and $y_i = [y_{i1}, y_{i2}, \dots, y_{in}] \in \mathbb{R}^n$) are the i th input vectors and i th target vectors, respectively, and suppose there are L nodes in the single hidden layer; output matrix of the hidden layer can be defined as:

$$\mathbf{A} = \begin{bmatrix} \phi(w_1 \cdot x_1 + b_1) & \cdots & \phi(w_L \cdot x_1 + b_L) \\ \vdots & \ddots & \vdots \\ \phi(w_1 \cdot x_H + b_1) & \cdots & \phi(w_L \cdot x_H + b_L) \end{bmatrix}_{H \times L}, \quad (4.1)$$

where $\phi(\cdot)$ is the activation function, and usually, a sigmoid function can achieve a relatively good result. $\mathbf{w}_l = [w_{l1}, w_{l2}, \dots, w_{lm}]^T$ and $\mathbf{b}_l = [b_1, b_2, \dots, b_L]^T$, $l = 1, \dots, L$, are randomly assigned weights and biases between the input nodes and the hidden layer before training. Say β_l is the weight connecting the l th hidden layer and output nodes, defined as:

$$\beta_{ELM} = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix} = \begin{bmatrix} \beta_{11} & \cdots & \beta_{1n} \\ \vdots & \ddots & \vdots \\ \beta_{L1} & \cdots & \beta_{Ln} \end{bmatrix}_{L \times n}. \quad (4.2)$$

To learn the parameter matrix of β_{ELM} with a dimension of $L \times n$, the ridge loss function is widely adopted as:

$$\arg \min_{\beta \in \mathbb{R}^{L \times n}} \|\mathbf{A}\beta_{ELM} - \mathbf{Y}\|^2 + \lambda_{ELM} \|\beta_{ELM}\|^2 \quad (4.3)$$

where \mathbf{A} is the matrix composed of the activation functions with dimensions $H \times L$; \mathbf{Y} is a matrix with dimensions $H \times n$ containing GT data:

$$\mathbf{Y} = \begin{bmatrix} y_1^T \\ \vdots \\ y_H^T \end{bmatrix} = \begin{bmatrix} y_{11} & \cdots & y_{1n} \\ \vdots & \ddots & \vdots \\ y_{H1} & \cdots & y_{Hn} \end{bmatrix}_{H \times n} . \quad (4.4)$$

Through solving the loss function, we can obtain the matrix β by:

$$\hat{\beta}_{ELM} = (\mathbf{A}^T \mathbf{A} + \lambda_{ELM} \mathbf{I})^{-1} \mathbf{A}^T \mathbf{Y} \quad (4.5)$$

where \mathbf{I} is an identity matrix with dimensions $L \times L$, the hyper-parameter λ_{ELM} helps obtain a reliable result when the matrix $(\mathbf{A}^T \mathbf{A} + \lambda_{ELM} \mathbf{I})$ is not full rank.

4.3 ELM Network Architecture

The structure of ELM is depicted in Figure 4.1. Suppose the input vector is a pixel-wise histogram measured by a TCSPC system containing 256 time bins in the inference phase. The number of output nodes depends on the number of life-time components we defined in synthetic datasets. For instance, if the measured data consists of bi-exponential decay model, the output layer should be configured as three nodes, namely, τ_1 , τ_2 , and α . We can easily obtain average lifetimes from Equations 3.2 and 3.3. All the histograms from the sensor are fed into

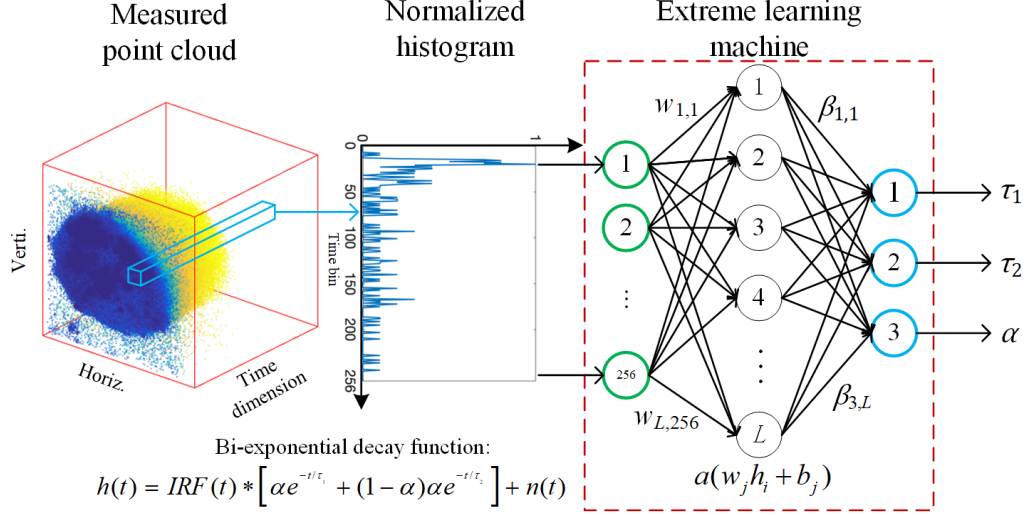


Figure 4.1: ELM is used for lifetime analysis. The input data are a 1-D pixel-wise histogram from the raw point cloud that contains 256 time bins. The histogram is fed into a single-hidden-layer ELM, and lifetime parameters (τ_1 , τ_2 , and α) can be obtained from output nodes.

the network sequentially; lifetime parameters can be obtained from output nodes pixel by pixel. The number of nodes in the hidden layer can be flexibly adjusted to achieve a trade-off between accuracy and computing time consumption.

4.4 Synthetic Data Analysis

τ_A and τ_I are used to estimate energy transfer for FRET or indicate fluorescence quenching behaviours [59]. This section compares NLSF, BCMM, and ELM to retrieve τ_A in bi-exponential decays. Likewise, we also compared NLSF, CMM, and ELM to reconstruct τ_I . Besides, ELM was compared with existing ANNs for FLIM in terms of (1) the network scale and (2) training time. Multiple

widely used metrics (F -value, SSIM, R^2 , MSE) were adopted for performance evaluations. Synthetic training datasets were generated with the same optical parameters using Equation 3.4 and 3.1, and Table 3.2.

4.4.1 Comparisons of Individual Lifetime Components

As NLSF was usually adopted in previous studies [60, 64, 66], we compared the inference performances of ELM and deconvolution-based NLSF (implemented with *lsqcurvefit*(\cdot) function in MATLAB[®] using iterative Levenberg–Marquardt algorithm) in Figure 4.2. As such, 2,000 simulated testing datasets were generated for recovery for single and double lifetimes. Here, we define the absolute error $\Delta g = |g - g_{est}|$, where $g = \tau_1, \tau_2, \alpha, \tau_A$ and g_{est} is the estimated g . Δg_{ELM} and Δg_{NLSF} are the absolute errors for ELM and NLSF. Figure 4.2 (a) and (b) show the Δg of ELM and NLSF for mono-exponential decays, respectively. Δg decreases as the peak intensity increases, and Δg_{ELM} is smaller than Δg_{NLSF} . Likewise, Figure 4.2 (c) and (d) indicate Δg plots for $g=\tau_1, \tau_2$, and α , where Δg_{ELM} is smaller than Δg_{NLSF} . Similarity, Figure 4.2 (e) and (f) indicate ELM obtained a much more accurate τ_A than NLSF. Therefore, ELM can perform better than NLSF in mono- and bi-exponential decays.

Additionally, as shown in Figure 4.3, we visually inspected estimate τ_1, τ_2 , and α , based on pre-defined variables in synthetic 2D images. We used the SSIM to evaluate reconstructed images in Figure 4.3 (a) and (b). The 2D lifetime images were reconstructed from a 3D synthetic data cube, composed of either mono- or bi-exponential decay ($256 \times 256 \times 256$, representing spatial and temporal dimensions). All the GT lifetime parameters (τ and α) are pre-defined in Equation 3.1. The 2D lifetime images are recovered pixel by pixel from noisy synthetic 3D data cubes. Figure 4.3 (a) shows reconstructed 2D images from mono-exponential

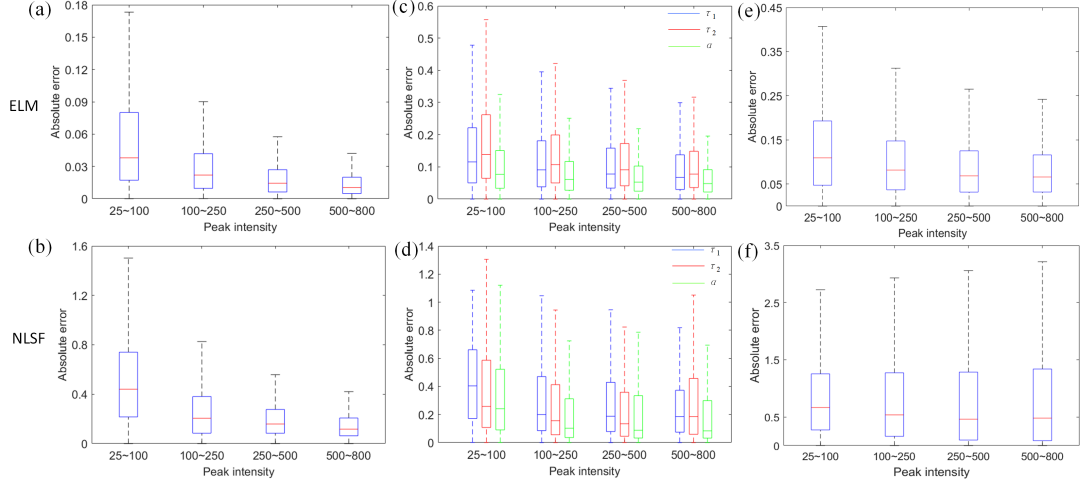


Figure 4.2: Box plots of absolute error versus different peak intensity levels regarding testing datasets. (a,b) Single lifetime estimations of mono-exponential decays from ELM and NLSF, respectively. (c,d) Double lifetime estimations of bi-exponential decays from ELM and NLSF, respectively. (e,f) τ_A estimated by ELM and NLSF, respectively.

decays with GT τ varying from 0.1 to 0.5 ns. Likely, Figure 4.3 (c) shows estimated τ_1 , τ_2 , and α bi-exponential decays. Results obtained from ELM are more accurate than NLSF. Figure 4.3 (c) shows the phasor plots of GT distribution of mono- (Figure 4.3 (a)), and bi-exponential (Figure 4.3 (b)) decays. From the phasor theory [95], cluster points of mono-exponential decays should locate on the semi-circle. For bi-exponential decays, two-lifetime components are indicated by the intersections of a fitted line and semi-circle. We utilize R^2 defined as

$$R^2 = 1 - \frac{\sum_{i=1}^P (\tau_A^i - \tau_{A_GT}^i)}{\sum_{i=1}^P (\tau_A^i - \tau_{A_Ave}^i)}, \quad (4.6)$$

to evaluate the estimation consistency, where τ_A^i is the predicted parameter $\tau_{A_GT}^i$ is the GT parameter $\tau_{A_Ave}^i$ is the average of GT parameters, P is the number of simulated decay curves. As shown in Figure 4.3 (d), scatter plots show ELM

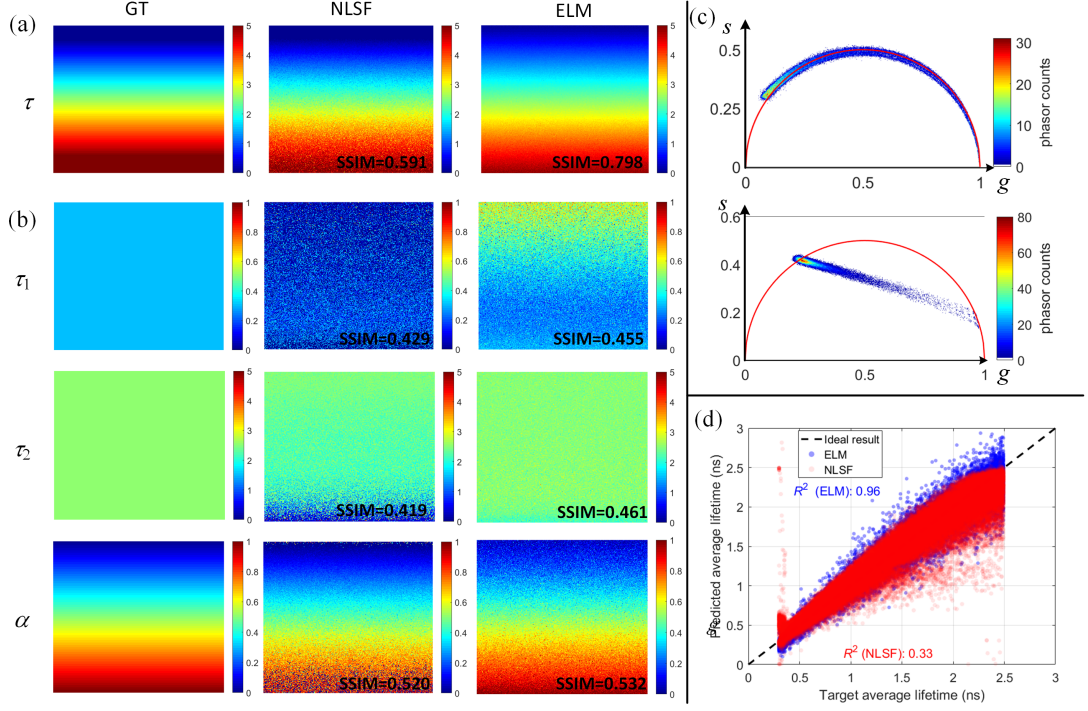


Figure 4.3: Lifetime parameters' estimation results, photon counts for each pixel were randomly picked between 25 and 500. (a) The estimated single lifetime using a mono-exponential decay model, where $\tau \in [0.1, 5]$ ns from top to down in the image. (b) The two estimated lifetimes using a bi-exponential decay model where $\tau_1=0.3$ ns, $\tau_2=3$ ns, and $\alpha \in [0, 1]$, from the top down. (c) Two phasor plots of GT distributions of (a, b). (d) Prediction accuracy and R^2 of τ_A from ELM and NLSF, with $\tau_1=0.3$ ns, and $\tau_2=2.5$ ns, respectively.

is closer to GT, and NLSF shows more outliers. We further evaluated ELM and NLSF using the F -value defined as Equation 4.7 with synthetic mono- and bi-exponential decays.

$$F = \frac{\delta x}{x} \cdot \sqrt{I}. \quad (4.7)$$

$F > 1$ and lower F means higher precision, where I is the detected photon count, δx is the standard deviation of the estimated lifetime parameter, and x is the GT parameter. We generated 200 synthetic decays for given ranges of lifetimes and peak intensities in Figure 4.4. Figure 4.4 (a) shows the F -value of mono-exponential decays versus the lifetime in the range [0.1, 5] ns. Figure 4.4 (b) shows the F -value of bi-exponential decays versus τ_1 , τ_2 , and α in [0.1,1] ns, [1, 3] ns, and [0, 1], respectively. We assigned 200 decays with a total photon count (<2,000) per synthetic histogram for both scenarios. Both figures show that ELM obtained a smaller F than NLSF, meaning ELM can achieve better precision. Furthermore, we defined the bias $\frac{\Delta\tau}{\tau}$ to evaluate ELM and NLSF versus the photon count. τ was set to 3.0 ns for mono-exponential decays. τ_1 , τ_2 , and α were set to 0.3 ns, 3.0 ns, and 0.5 for bi-exponential decays. Figure 4.4 (c) shows that the bias of NLSF increases as the photon count increases, which is worse than ELM. Figure 4.4 (d) shows that the bias of ELM is smaller than NLSF, and ELM is more robust to varying photon counts. Moreover, NLSF is also sensitive to initial conditions of lifetime parameters [59]. The bias decreases when the initial conditions are closed to GT values, meaning that users need to have prior knowledge about the parameters to be extracted.

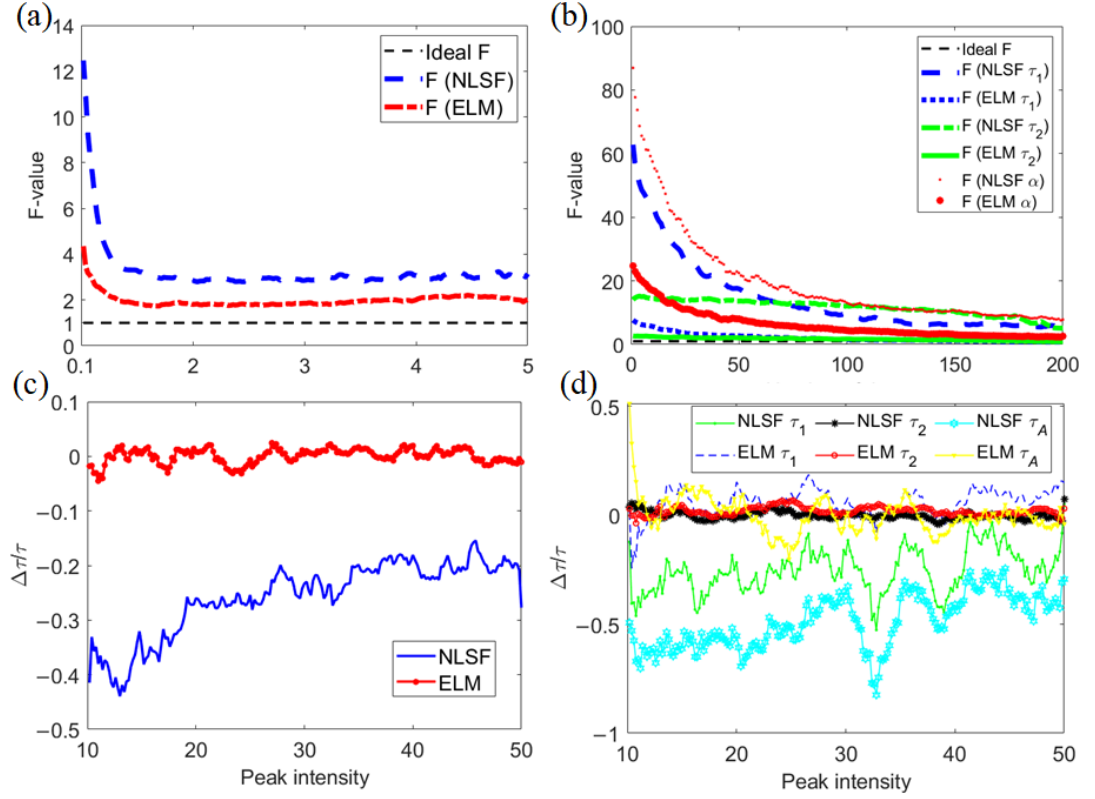


Figure 4.4: F -value vs with τ_1 , τ_2 , and α in the ranges $[0.1, 1]$ ns, $[1, 3]$ ns, and $[0, 1]$, respectively. (c,d) Bias per histogram for mono- and bi-exponential decays.

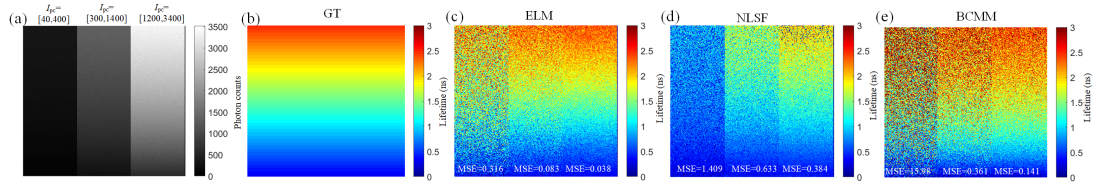


Figure 4.5: (a) Intensity image of GT τ_A in n exact ranges. I_{pc} depicts total photon counts in one pixel. The range from 40 to 400 is viewed as low photon counts. (b) the GT τ_A lifetime image with the range $[0.3, 2.5]$ ns. (c-e), τ_A images from ELM, NLSF, and BCMM.

Table 4.1: Time Consumption (Seconds) of NLSF and ELM for Reconstructing Life-time Parameters.

Algorithm	Mono-Exponential mode	Bi-Exponential mode
NLSF	371.9 s	670.9 s
ELM	6.2	6.5
CMM [3]	1.9	1.9 (τ_I)
BCMM [57]	-	16.1(τ_A)

4.4.2 Comparisons of τ_A

We evaluated ELM in estimating τ_A in various count conditions. As shown in Figure 4.5 (a), we set three regions at three count levels, changing τ_A from top to bottom. We refer to the three regions as low, middle, and high counts hereafter. Figure 4.5 (b) depicts the GT τ_A . From Figure 4.5 (c) and (d), ELM shows a more accurate τ_A image than NLSF, with ELM producing a smaller MSE than NLSF in each region. We also included the non-fitting BCMM [57], for the comparison due to its fast speed and capacity to resolve bi-exponential decays. From Figure 4.5 (e), BCMM is not robust in low counts, outperforming NLSF in middle and high regions. Further, ELM obtained better results than BCMM. BCMM is less photon efficient, and it is sensitive to the measurement window T (T should be larger than $5 \times \tau_2$, otherwise bias correction is needed [57]). Table 4.1 compares ELM with NLSF regarding the time consumption for inference (forward-propagation) tasks in Figure 4.3 (a) and (b). NLSF resolving mono-exponential decays consumes more time than for bi-exponential decay models. In contrast, the analysis time of ELM is not affected by the number of lifetime components and it is substantially less than NLSF.

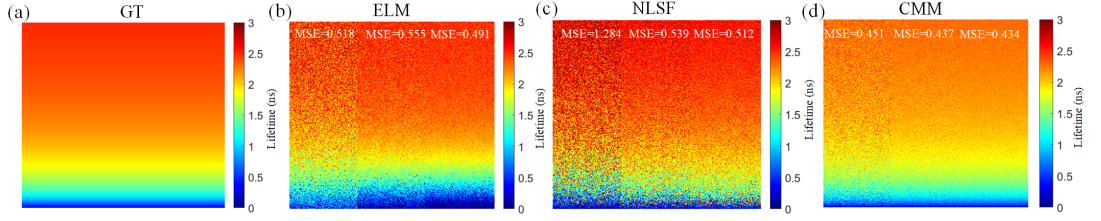


Figure 4.6: (a) GT τ_I image in exact ranges. (b-d), Reconstructed τ_I images from ELM, NLSF, and CMM for bi-exponential decays.

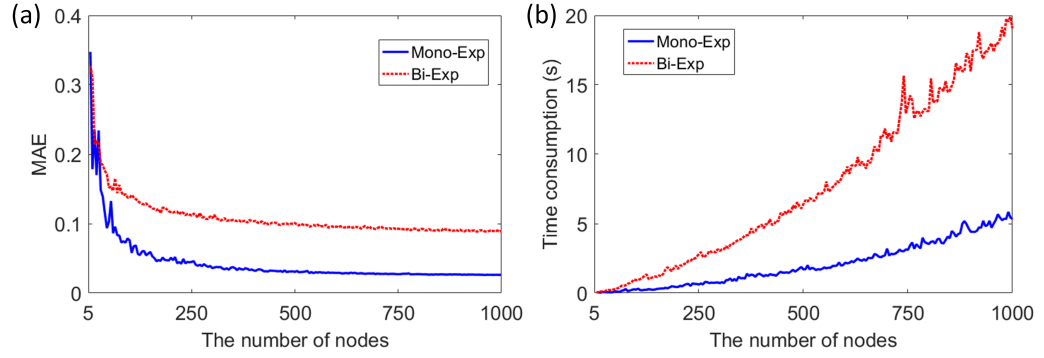


Figure 4.7: (a), (b) Loss curves and time consumption vs. different numbers of nodes in the hidden layer.

Table 4.2: Comparisons of Existing NN Architecture for Lifetime Estimation.

Algorithm	Training Parameters	Hidden Layer	Resolve Multi-Exp. Decays	Training Time
ELM	205 600	1	✓	10.85 s
FLI-NET [64]	1 084 045	7	✓	4 h
1D CNN [62]	48 675	7	✓	23 min
MLP [4]	3 750 205	3	×	38 min
MLP [60]	149 252	2	✓	4 h

4.4.3 Comparisons of τ_I

CMM [3] achieves the fastest speed for intensity average lifetime analysis. We further compared CMM with ELM for τ_I reconstruction. As shown in Figure 4.6 the result from ELM is better than NLSF but slightly worse than CMM. However, CMM is sensitive to and biased by measurement window if bias correction is not included. Although CMM obtained a smaller overall MSE, the bias occurs as τ_I becomes longer. It agrees with the conclusion from the previous work [59], indicating that CMM causes misleading inference when there are multi-lifetime species in the field of view. Further, τ_I sometimes generates a shorter dynamic lifetime range than τ_A as τ_I cannot correctly distinguish clusters with different lifetimes, especially for strong FRET phenomena [96]. ELM and CMM can achieve shorter processing time than NLSF and BCMM, as shown in Table 4.1. In this case, although ELM is slightly slower than CMM, the consumed time varies with the number of nodes in the hidden layer. Figure 4.7 (a) shows training errors indicated by MAE versus different numbers of nodes in the hidden layer. Here, the number of the hidden layer is set to 500 for both mono- and bi-exponential models, as there was no apparent MAE decrease, and a moderate processing time

was achieved, as shown in Figure 4.7 (b). Moreover, we compared ELM with relevant ANNs for FLIM. Since ELM uses the Moore–Penrose matrix inversion strategy to learn parameters instead of back-propagation, it is much faster. As shown in Table 4.2, although ELM has more parameters than 1D CNN [62], the training time is much shorter than the other existing studies [4, 60, 62, 64]. Many CNN hyperparameters should be fine-tuned, and batch normalizations should be implemented to avoid gradient vanishing [97]. In contrast, ELM’s architecture is much simpler, and we simply need to adjust the number of nodes in the hidden layer. Furthermore, the efficient training process enables online training and is suitable for embedded hardware implementations [98]. ELM is highly reconfigurable to provide a flexible solution to balance the trade-off between computing complexity and accuracy. The evaluations of ELM and NLSF were conducted in MATLAB[®] R2016a, 64-bit CPU (Intel Core i5-4200H @ 2.80 GHz) with 8 GB memory. Notably, other studies in Table 4.2 used much more powerful GPU to train their models. Despite this, ELM still delivers the shortest training time.

Based on the analysis of synthetic datasets, ELM is more robust for analyzing mono- and bi-exponential decays than traditional NLSF methods. We will evaluate ELM using realistic experiment data in the next section.

4.5 Experimental FLIM Data Analysis

To investigate the feasibility of ELM for experimental FLIM data, we utilized living prostate cancer cells incubated with functionalized Gold Nanorods (GNRs). A commercial two-photon FLIM system was used to acquire raw 3D data cubes. This section compares ELM with 1D-CNN, NLSF, and BCMM.

4.5.1 Experimental Setup and Sample Preparation

We used the proposed ELM to analyze a living cellular sample, acquired by a two-photon FLIM system. To achieve an efficient imaging contrast, prostate cancer cells were treated with GNRs functionalized with Cy5 labeled ssDNA [99]. GNRs have tunable longitudinal surface plasmon resonance and enable the interactions between the strong electromagnetic field and activated fluorophore in biological samples [100, 101]. Functionalizing GNRs with fluorophore-labelled DNA has been adopted to probe endocellular components [102, 103], including microRNA detections for human breast cancer or monitoring the intracellular level of metal ions in human serums. Here, prostate cancer cells were incubated with nanoprobe for 6 hours and washed three times with phosphate-buffered saline. Cells were blended with 4% paraformaldehyde for 15 min. After removing paraformaldehyde, cells were washed with distilled water three times. The two-photon FLIM platform consists of a confocal microscope (LSM 510, Carl Zeiss, Oberkochen, Germany) with 256×256 spatial resolution, where the scan module includes four individual PMTs. A TCSPC module (SPC-830, Becker & Hickl GmbH, Berlin, Germany) with 256 time bins and 39 picosecond timing resolution was mounted on the microscope. A tunable femtosecond Ti: sapphire laser (Chameleon, Coherent, Santa Clara, CA, USA) was configured with a repetition frequency 80 MHz and 850 nm wavelength to excite the sample. The emission light was collected using a $60 \times$ water-immersion objectives lens (numerical aperture = 1.0) and a 500–550 nm bandpass filter. One hundred scanning cycles were selected to prevent GNRs heating and obtain sufficient photons, where each cycle took three seconds.

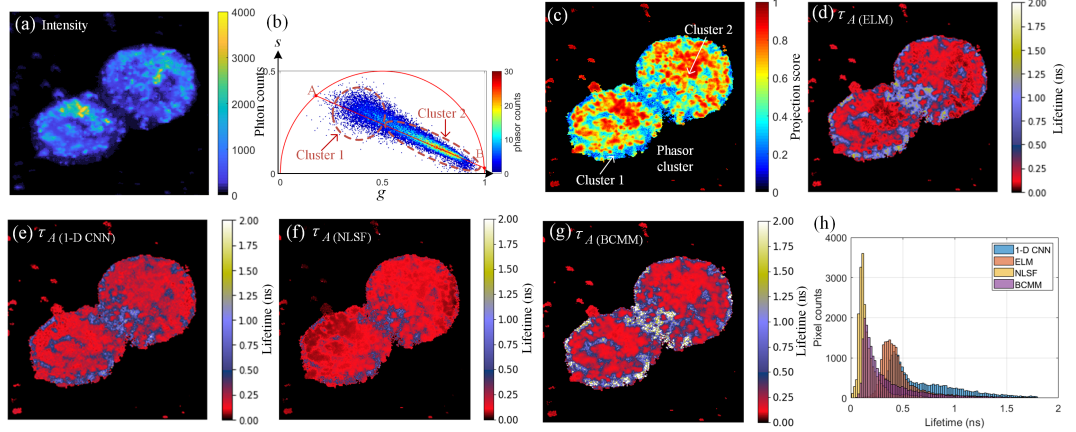


Figure 4.8: Lifetime analysis of prostatic cells loaded with gold nanoprobes. (a) The intensity image, (b) phasor plot, and (c) phasor projection image. (d–g) τ_A restored by ELM, 1D CNN, NLSF, and CMM. (h) Lifetime histograms of ELM, 1-D CNN, NLSF, and BCMM.

4.5.2 Algorithm Evaluation

Due to the strong two-photon photoluminescence property of GNRs, high optical discernibility can be observed between the GNRs and cell tissues [104]. Figure 4.8 (e), shows the grey-scale intensity image of the sample, where the bright spots are GNRs. As the background pixels with fewer photon counts imply less useful information, they can be neglected during the analysis. In this case, a threshold (100 photon counts) was considered to neglect these pixels. As conventional data readout from TCSPC systems is pixel by pixel, accumulated histograms can be directly fed into the ELM without data conversion. The biological sample should be illuminated with a long acquisition time to achieve a high SNR to obtain a reliable reference. However, a long acquisition time can easily lead to photobleaching. The previous study [62], reported that a phasor projection image could alternatively serve as a reference image to identify autofluorescence and gold nanoprobes. Two clusters representing autofluorescence of

the cell and gold nanoprobe can be observed in the phasor plot shown in Figure 8b, after we had applied pixel filtering. Cluster 2 contains the majority of pixels with shorter lifetimes depicting gold nanoprobe. A fitted line was obtained by a linear regression fitting algorithm:

$$\arg \min_{a,b} \sum_{n=1}^M ||s_n - (ag_n + b)||_2^2 \quad (4.8)$$

where a and b are slope and intercept of the fitted line, g_n and s_n are locations of pixels in the phasor domain. The intersection points $A(g_a, s_a)$ and $B(g_b, s_b)$ can be obtained accordingly. As shown in Figure 4.8 (c), we employed the pixel-wise phasor score ρ to generate a phasor projection image by computing

$$\rho_n = [(g_n - g_2)(g_1 - g_2) + (s_n - s_2)(s_1 - s_2)]/D \quad (4.9)$$

where D is the Euclidean distance between A and B , n is the number of filtered pixels.

By comparing τ_A images obtained from ELM (Figure 4.8 (d)), 1D CNN (Figure 4.8 (e)), NLSF (Figure 4.8 (f)), and BCMM (Figure 4.8 (g)), the image from NLSF shows obvious bias because, as mentioned, NLSF is sensitive to initial values and fails to converge sometimes. Given that the 1D CNN [62] achieved high speed and accuracy, we compared ELM and the 1D CNN in term of τ_A using the same training datasets. From Figure 4.8 (d), (e), ELM is in good agreement with the 1D CNN, and they showed similar distributions of pixel counts, as shown in Figure 4.8 (g), the NLSF's result is significantly more biased than the other three algorithms. This is because deconvolution was involved in NLSF, causing non-convergent results due to dealing with ultra-short decays caused by

gold nanoprobe. As mentioned, BCMM is not robust in varying ranges of photon counts; many pixels are out of the defined range (0 to 2 ns), as the white pixels show in Figure 4.8 (g). Nevertheless, BCMM is a fast algorithm that only took 6.53 s to reconstruct the image. The inference time of 1D CNN on a GPU (NVIDIA GTX 850M) is 116.43 s, whereas ELM only consumed 1.73 s during inference on the CPU.

4.5.3 Low Counts Scenarios

Fragile tissues, such as retinas, cannot be excited by laser for a long time. To avoid tissue damage and photobleaching caused by a long acquisition time, we investigated ELM's performance for data in low-photon scenarios. We kept the experimental setup identical to Section 4.5.1. To acquire less-emitted photons, we chose the field of view with fewer nanoprobe. Increased scanning cycles were set on the software. As the number of cycles increased, we changed the intensity threshold to guarantee sufficient pixels were saved. The value of the intensity threshold should be fine-tuned according to different bio-samples (5% of total counts in our experiments). Figure 4.9 (a), (b) depict intensity and reconstructed τ_A images, respectively. The lifetime of cells and nanoprobe can be consistently reconstructed, even if the cycle decreases to 10. Notably, nanoprobe and boundaries of cells cannot be identified in intensity images with 10 and 40 cycles, yet lifetime images can restore the lifetime and reveal cell boundaries. Below each lifetime image in Figure 4.9 (b), histograms of pixel occurrence were below τ_A images, showing means μ and standard deviations δ . There was no distinct shift in μ and δ at different collection cycles, indicating that ELM is robust, even at low counts.

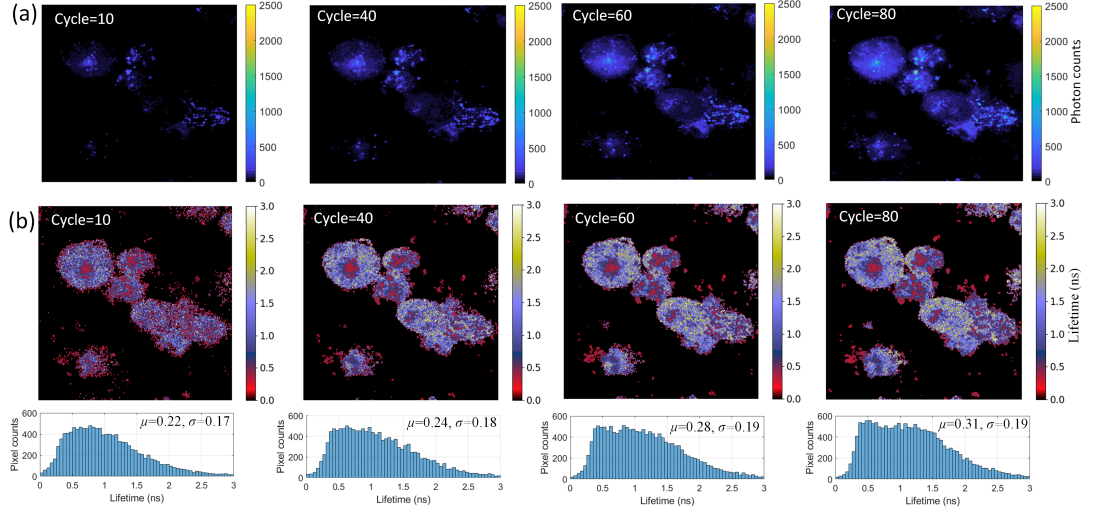


Figure 4.9: (a) Intensity images with different scales of colorbars, scanning cycles were set to 10, 40, 60, and 80. Colorbars are unified. (b) τ_A ages and pixel occurrence reconstructed by ELM in different cycles.

4.6 Summary and Future Work

In summary, we presented an ELM architecture to accurately retrieve fluorescence lifetime parameters from mono- and bi-exponential decays. Both synthetic and realistic experimental FLIM datasets were employed to evaluate the proposed network. Our results show ELM outperforms fitting and non-fitting methods, regarding synthetic datasets at different photon counts. Further, ELM can better identify GNRs and cells and yield a comparable result to the 1D CNN method. Since ELM does not need back-propagation to train the network, it is more flexible to reconfigure the network topology. Due to the potential on-line training property, it is promising to implement it on embedded hardware in the future, coupling with sensors and readout circuits to achieve fast on-chip training and

inference. Despite histogram data from different optical instruments, the on-line training will enhance the generalization. More FLIM applications relying on GNRs will benefit from this study for cellular cancer diagnosis.

Chapter 5

Towards High-performance Deep Learning Architecture and Hardware Accelerator Design for Robust Parameters Analysis in Diffuse Correlation Spectroscopy

5.1 Background

Blood flow is a critical bio-indicator to investigate the consumption and supplement of oxygen and glucose in the brain and muscles. Existing blood flow sensing techniques have been summarized in previous studies [105], [106], among which DCS is making a remarkable stride in monitoring cerebral [107], [5] and muscular [108], [109] blood flow variations in non-invasive, continuous manners. In essence, DCS measures how fast coherent light loses coherence because of the movement of red blood cells. DCS uses a near-infrared laser and a photon-sensitive detector (such as SPAD [5], [91], [6, 110, 111], avalanche photodiodes [105], [112], PMT [113]) placed near the laser with a SDD. After the laser illu-

minates the tissue, the detector collects scattered photons. The detected intensity fluctuations are then fed into correlator hardware to compute the intensity auto-correlation function ACF, i.e., $g_2(\tau)$. $g_2(\tau)$ is related to the electric field autocorrelation function, $g_1(\tau)$, defined by the Siegert relation [114]. The optical parameters μ_a and μ'_s , BFi, and the coherence factor β can be retrieved by fitting measured $g_2(\tau)$ via analytical models defining the solution of correlation diffusion equations. Fitting algorithms [91], [11] are suitable for single-point detectors in terms of accuracy and speed. However, advanced CMOS SPAD arrays are emerging DCS detectors, as parallelized acquisition generates a higher SNR compared with single-point detectors. Massively parallelized acquisition increases data throughput and requires efficient algorithms and hardware architectures to interpret DCS data. Despite the efficacy of conventional fitting. Despite the efficacy of conventional fitting [11], [115], [116], [117] and DNNs [118], [119], [120], [121] for either single-point detectors or SPAD arrays, their speeds are not applicable for array detectors due to the high throughput. Furthermore, a highly integrated hardware computing architecture is necessary for the miniaturization of DCS systems. We target the challenges mentioned and propose strategies in four aspects.

1. We propose an ACNN tailored for hardware implementation, focusing on reconstructing BFi and β from intensity ACFs. The ACNN architecture utilizes multiplication-free convolutions to alleviate computational complexity, enabling higher parallelism and lower hardware utilization. Notably, addition operations save half of the latency compared with multiplication operations according to micro-instructions of various CPU operations [122].

2. To assess ACNN’s performance in characterizing BFi, we quantitatively compare speed and accuracy with a conventional CNN that performs the same network topology and training strategies. This evaluation is based on a semi-infinite analytical model and *in-silico* MC simulations of the single-layer model of milk with known diffuse parameters. Our results demonstrate that ACNN achieves accurate reconstructed BFi in the inference phase.
3. We present a heterogeneous computing platform implemented on Zynq-7000 and UltraScale+ multiprocessor systems-on-chip (MPSoC) FPGA. The ACNN accelerator is implemented on FPGAs. The ACNN accelerator fully explores the parallelism of the ACNN model, achieving different levels of parallelism ranging from nested for-loops unrolling to pixel-wise parallelism. Alongside FPGA fabric, SIMD on the embedded CPU is enabled to accelerate ACF generation.
4. To further miniaturize the accelerator, we employ various quantization strategies with different bit widths. Herein, we report the corresponding hardware utilization and speeds on cost-effective and high-performance FPGAs. This analysis examines the trade-off between reconstruction accuracy and hardware efficiency, facilitating choosing application-specific configurations aiding in selecting appropriate configurations for specific application needs.

The following sections are organized: Section 5.2 provides a comprehensive review of relevant literature, highlighting advancements in reconstruction algorithms and on-chip processing methods. Prospective enhancements in these areas are also presented. Section 5.3 illustrates the DCS theory for generating ACNN’s training datasets. A canonical MC simulation was used as the reference to validate the

consistency with the analytical model. Section 5.4 presents a detailed description of our ACNN and hardware implementation, and quantitative analysis. Section 5.5 summarizes this study and indicates future work.

5.2 Prior Work

This section reviews existing algorithms for reconstructing BFi and β from ACFs. Besides, state-of-the-art on-chip processing strategies are also examined. We illustrate potential improvements in the two aspects of our work targets.

5.2.1 Algorithms Review

Existing BFi reconstruction algorithms can be categorized into two streams: optimization fitting algorithms and deep-learning approaches. Fitting measured ACF with analytical models to extract BFi and β is an ill-posed regression problem. The Matlab NLSF (Mathwork, Inc., USA) functions, for example, *lsqnonlin*(\cdot) using the interior-reflective Newton method [116], *fminsearch*(\cdot), the Nelder-Mead simplex algorithm [120] and *optimset*(\cdot) the Levenberg Marquardt method [123], have been adopted to reconstruct BFi and to assess errors resulting from uncertainties in optical properties and tissue thicknesses. Given that the fitting methods are constrained optimization problems, they involve numerous iterations and are therefore time-consuming. This time-consuming problem can be remarkably mitigated using data-driven DNNs that perform fast forward-propagation in inference. DNNs have catalyzed improvements in enhancing reconstruction and accuracy. For example, intensity ACFs were first converted into 2-D images and fed into a 2-D CNN for BFI and β reconstruction [119], achieving a 23-fold speedup compared to a nonlinear fitting method. Another study pro-

posed a LSTM for BFi reconstruction and relative blood flow analysis due to its superior capability for extracting features from sequences of data [120]. Similarly, gated recurrent units were embedded with a 1-D CNN to enhance information extraction, thereby retrieving relative BFi [121]. An LSTM variant [118] was proposed to denoise ACFs and extract BFi. Despite existing DNNs' high accuracy for DCS, redundant trainable parameters and complex topologies impede on-chip, real-time processing. The motivation to design a compact DNN is that most modern CW and time-domain DCS systems [5], [124] use SPAD arrays for data acquisition, where FPGAs are essential for controlling clocks and decoding data. We are inspired to embed the analysis on-chip to achieve end-to-end processing, taking the frame-based intensity as input and generating BFi and β .

5.2.2 Hardware Processor Review

Researchers have successfully implemented on-FPGA autocorrelators and on-FPGA BFi reconstruction. Buchholz *et al.* implemented a multi-channel autocorrelator for a 32×32 SPAD array, but the normalization of pixel-wise ACFs was not implemented [125]. To alleviate the computational burden, F. Rocca *et al.* [6] proposed an on-chip, scalable column-wise autocorrelator that can simultaneously compute up to 128 columns for a SPAD array with 192×64 enabled pixels. Another study [110] employed two FPGAs to accumulate detected photons and compute ACF for a 500×500 SPAD array, implementing element-wise matrix multiplications. But computationally expensive divisions and square root operations for FPGAs are implemented on PCs.

Besides embedded autocorrelators, an iterative nonlinear curve-fitting algorithm was implemented on-FPGA using LabVIEW [117]. Although it achieves real-time BFi reconstruction, the high-level LabVIEW implementation exhibits

a coarse control over allocating logic and data paths, leading to a redundant hardware overhead. Also, iterative operations significantly hinder on-FPGA data pipeline, thereby deteriorating the throughput. Overall, existing FPGA platforms merely integrate ACF generation and BFi reconstruction in a monolithic fashion. In this study, we take full advantage of reconfigurable heterogeneous SoC platforms embedding CPUs and PL, to encapsulate all computing pipelines on-chip, including ACF generation and DL-accelerators for BFi and β reconstruction. The proposed approach demonstrates superior efficiency compared to a common CPU and GPU. Extended from our previous work [126], [79] for an FPGA-embedded DL processor for fluorescence lifetime imaging, we proposed a more concise, multiplication-free, CNN for estimating BFi and β .

5.3 Analytical Model

This section introduces a generic DCS analytical diffusion model, adopted for synthetic dataset generation hereafter. Besides, we adopted MC simulations to simulate a semi-infinite phantom to validate the consistency between analytical models and MC simulations. For typical DCS systems, ACF can be calculated by

$$g_2(r, \tau) = \frac{\langle I(r, t) \cdot I(r, t + \tau) \rangle}{\langle I(t) \rangle^2}, \quad (5.1)$$

where $I(r, t)$ indicates the photon intensity at time t and position r , $\langle \cdot \rangle$ means a time average, τ is the lag time. The electric field ACF $g_2(r, \tau)$, satisfied the CDE in a scattering tissue [127]

$$(D\nabla^2 - v\mu_a - 1/3v\mu'_s k_0^2 \alpha \langle \Delta r^2(\tau) \rangle) G_1(r, \tau) = -vS(r). \quad (5.2)$$

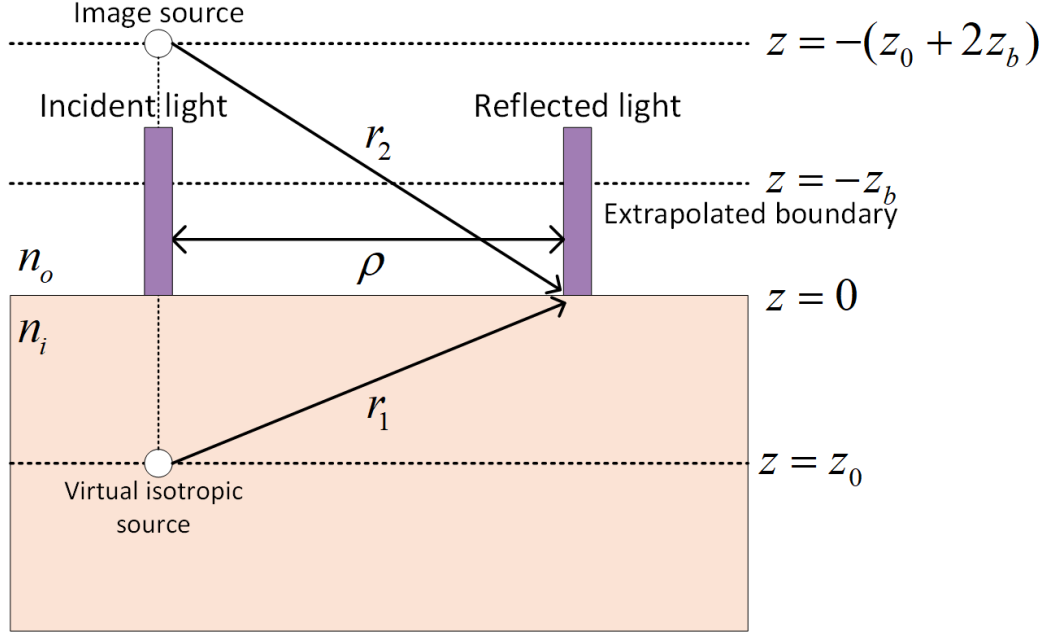


Figure 5.1: The concept of spatial diffuse reflectometry in a semi-infinite geometry.

In Eq. 5.2, $D = \frac{v}{3(\mu_a + \mu'_s)}$ denotes the photon diffusion coefficient, μ_a and μ'_s the absorption and reduced scattering coefficients, v the light speed in the medium, k the wavenumber in the medium, and α the ratio between dynamic scatters and all scatters. As the Brownian motion model has been widely adopted for specific biological tissues in DCS research [11], [127], [128], this work focuses on Brownian motions because Brownian motion model is adequate for most bio-tissue [112]. In future we can derive a version for the ballistic model [129]. The Brownian motion model $\langle \Delta r^2(\tau) \rangle$ describes the mean-square displacement, and $S(r)$ is the CW isotropic light source. The tissue can be modelled as a semi-infinite medium bounded by the tissue surface for biomedical tissues with a high-scattering property. And the solution [112], [11]. $G_1(r, \tau)$ in Eq. 5.1 can be

represented as

$$G_1(\tau) = \frac{3\mu'_s}{4\pi} \left(\frac{\exp(-Rr_1)}{r_1} - \frac{\exp(-Rr_2)}{r_2} \right), \quad (5.3)$$

where $R^2 = 3\mu'_s\mu_a + \alpha\mu_s'^2 k_0^2 \langle \Delta r^2(\tau) \rangle$, and $r_1 = \sqrt{\rho^2 + z_0^2}$ and $r_2 = \sqrt{\rho^2 + (z_0 + 2z_b)^2}$ are shown in Figure 5.1. Besides, $k_0 = \frac{2\pi}{\lambda}$, where λ is the wavelength in the medium. $R_n = \frac{n_i}{n_o}$, where n_i and n_o are the refractive indices inside and outside of the tissue. $z_0 = \frac{1}{\mu'_s}$ means the distance between the virtual isotropic point source and the tissue surface. $z_b = \frac{2(1+R_{eff})}{3\mu'_s(1-R_{eff})}$ means the distance between the extrapolated boundary and the tissue surface, where the effective reflection coefficient, $R_{eff} = -1.440R_n^{-2} + 0.710R_n^{-1} + 0.668 + 0.0636R_n$ indicates the internal reflection coefficient between two media. As we mentioned, $\langle \Delta r^2(\tau) \rangle$ can be approximated to $6D_B\tau$ for diffusive motion [130]. Therefore, R^2 can be derived as $3\mu'_s\mu_a + \alpha\mu_s'^2 k_0^2 6D_B\tau$. The details about spatial diffuse reflectometry have been explained in [131].

Recalling the Siegert relation [114], $g_2(\tau) = 1 + \beta |g_1(\tau)|^2$, $g_1(\tau) = G_1(\tau)/G_1(0)$, where β is the coherence factor, mainly determined by the system setup. It has been proven that αD_B can present BFi [107], [132], [133]. Also, for liquid phantom, $\alpha \approx 1$ [132]. Alongside generating $g_2(\tau)$ with the analytical model, the noise model of $g_2(\tau)$ is also crucial for simulating noise in real media. Existing studies [112], [11], [134], [135] state that the noise $\delta(\tau)$ (standard deviation) of measured $(g_1(\tau) - 1)$ can be approximated as

$$\delta(\tau) = \sqrt{\frac{T}{t}} \left[\beta^2 \frac{(1 + e^{-2\Gamma T})(1 + e^{-2\Gamma\tau}) + 2m(1 - e^{-2\Gamma T})e^{-2\Gamma\tau}}{1 - e^{-2\Gamma T}} + 2\langle n \rangle^{-1} \beta(1 + e^{-2\Gamma\tau}) + \langle n \rangle^{-2} (1 + \beta e^{-\Gamma\tau}) \right]^{1/2} \quad (5.4)$$

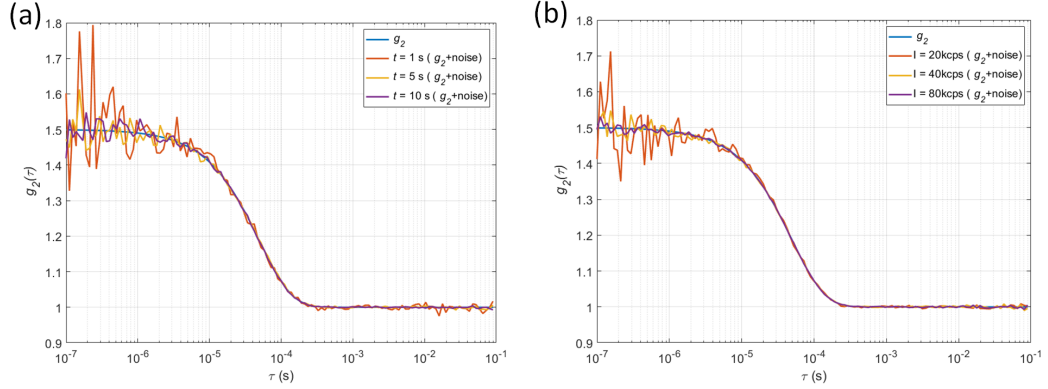


Figure 5.2: $g_2(\tau)$ curves with fixed optical parameters but with (a) different averaging time ($t = 1$ seconds, 5 seconds, and 10 seconds) and (b) with different photon intensities ($I = 20$ kcps, 40 kcps, and 80 kcps).

in a homogeneous medium with an infinite geometry. Here, T is the time intervals between two adjacent τ , m is the bin index, t is the total averaging time. $\langle n \rangle = I \cdot T$ the average number of photons, and I is the photon count rate. Γ is the decay rate of a single exponential function that approximates $g_1(\tau) = \exp(-\tau/\tau_c)$ [134]. Since τ_c is an unknown parameter, we apply the fitting method *fminsearch*(\cdot) in MATLAB[®] to retrieve it after we obtain $g_2(\tau)$ from the Siegert relation. We investigated the sensitivity in terms of the averaging time and photon intensity. We set the other optical parameters as constant, namely, we fixed μ_a and μ'_s to 0.1 mm^{-1} and 2.0 mm^{-1} , $\rho = 10 \text{ mm}$, $\text{BFi} = 5 \times 10^{-7} \text{ mm}^2/\text{s}$, $\beta = 0.5$, and $\lambda = 700 \text{ nm}$. As shown in Figure 5.2, the noise is related t , and $g_2(\tau)$ curves become noiser when t decreases. Similarly, the amplitude of noise is negatively proportional to I .

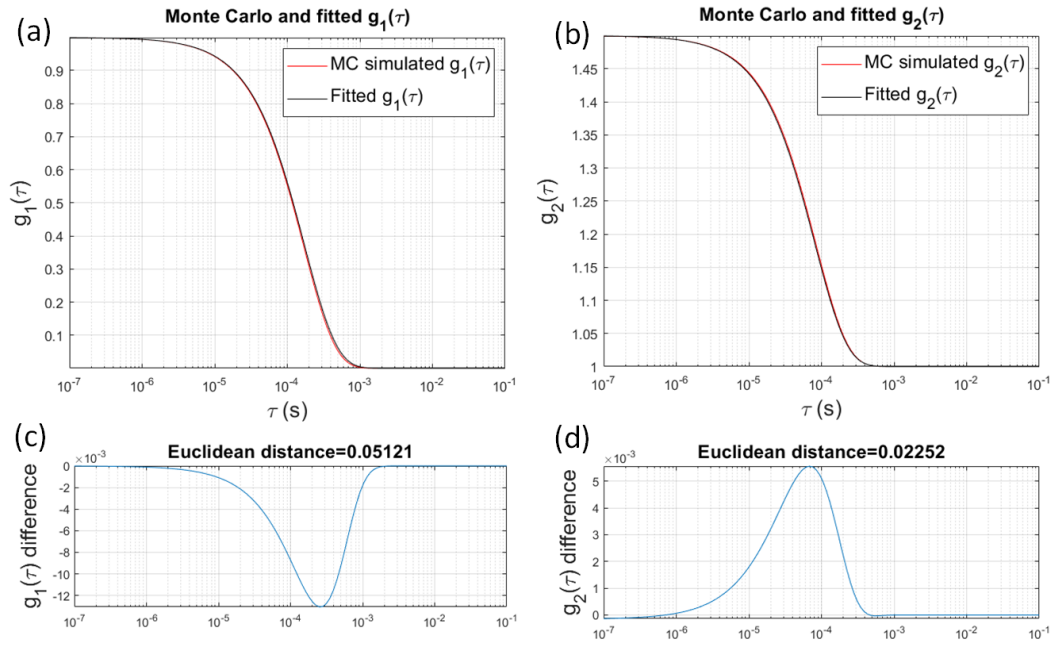


Figure 5.3: Comparison between MC simulation and analytic model for $g_1(\tau)$ and $g_2(\tau)$ of milk. (a) and (b) generated and fitted curves using MCX and analytical models. (c) and (d) Euclidian distance between simulated and fitted curves.

5.3.1 Monte Carlo Simulation

We chose the single-layer model of milk ($\mu_a = 0.27 \text{ mm}^{-1}$ and $\mu'_s = 160 \text{ mm}^{-1}$ at 780 nm [136]) in the Monte Carlo eXtreme (MCX) photon propagation simulations [115], [137]. We configure that 10^7 photons were emitted from a light source. The radius of the detector is 1 mm . ρ was configured to be 10 mm . The voxel of the phantom is 60 mm^3 . MCX employs *fminsearch*(\cdot) as the non-linear fitting algorithm by default. The fitting method can obtain accurate results because the simulated curves are noise-free. Once we obtained parsed $g_1(\tau)$ using MCX studio, we used Siegert relation to calculate the corresponding $g_2(\tau)$, depicted by red lines in Figure 5.3 (a) and (b), respectively. Also, by using analytical model, we obtained the fitted $g_1(\tau)$ and $g_2(\tau)$ with reconstructed αD (equivalent to BFi) and β , shown by black lines in Figure 5.3 (a) and (b). The Euclidian distances shown in Figure 5.3, (c) and (d) indicates small errors. Therefore, the reconstructed BFi and β can be the reference for evaluating our algorithms. Also, the analytical model achieves nearly consistent $g_1(\tau)$ and $g_2(\tau)$ curves compared to Monte Carlo simulations in MCX, meaning that we can quickly generate training datasets for our DNN model by constructing and automating the analytical model in MATLAB[®].

5.4 DL Architecture

Inspired by the previous 1D CNN FPGA implementation [126], [79], [138], we propose a similar but more compact DL network that does not involve multiplications apart from BNs. Furthermore, we do not use ResNet blocks [85] compared with the previous work [126] as its skip connections introduce data dependency that impedes data pipelining and for-loop unrolling on FPGAs. We have proven

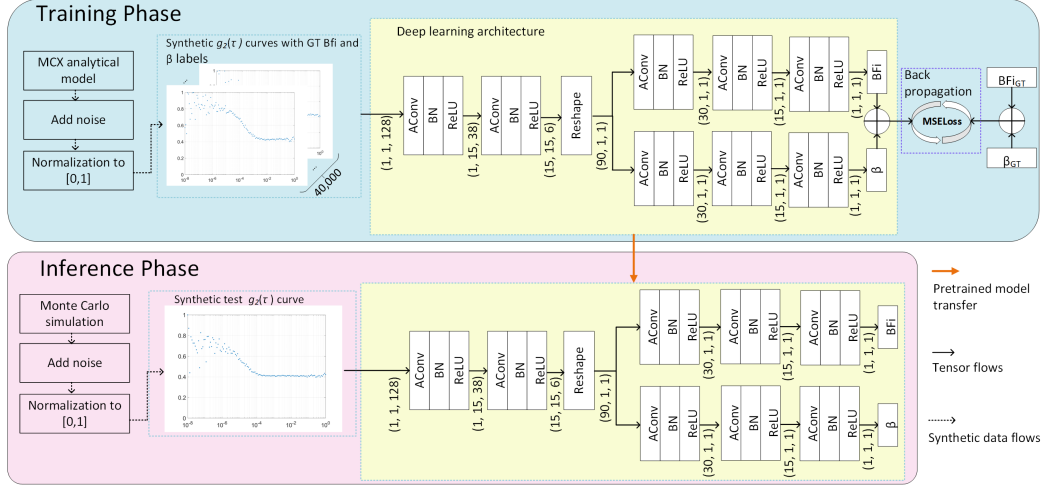


Figure 5.4: ACNN architecture in training and inference phases.

that our network can converge without ResNet blocks. The DL architecture is depicted in Figure 5.4. The $g_2(\tau)$ curves synthesized from the analytical model were applied to training and validation as the model is based on rigorous deductions and assumptions. On the other hand, using the curves generated from MC simulations for the test datasets provides a realistic evaluation of the model's performance in real-world scenarios.

We generated 40,000 $g_2(\tau)$ curves for training, using the analytical model (Eq. 5.3) and the noise generator (Eq. 5.4). The dataset is configured with $\beta \in [0, 1)$, $\text{BFi} \in [10^{-8}, 10^{-5}] \text{ mm}^2/\text{s}$, $\rho = 10 \text{ mm}$, and $\lambda = 700 \text{ nm}$, to emulate realistic experiments. The noise rate varies from 20 kcps to 80 kcps. 10% (4,000) curves of the training dataset were used for validation during training. The optimizer is RMSprop, and the Huber loss was adopted as the loss function. The learning rate is 0.001 with 0.997 exponential decreasing. It took 18 minutes for the NVIDIA RTX A1000 GPU to train the model. 35 patience epochs were used to avoid overfitting. The total number of training epochs is 300. All $g_2(\tau)$ datasets included

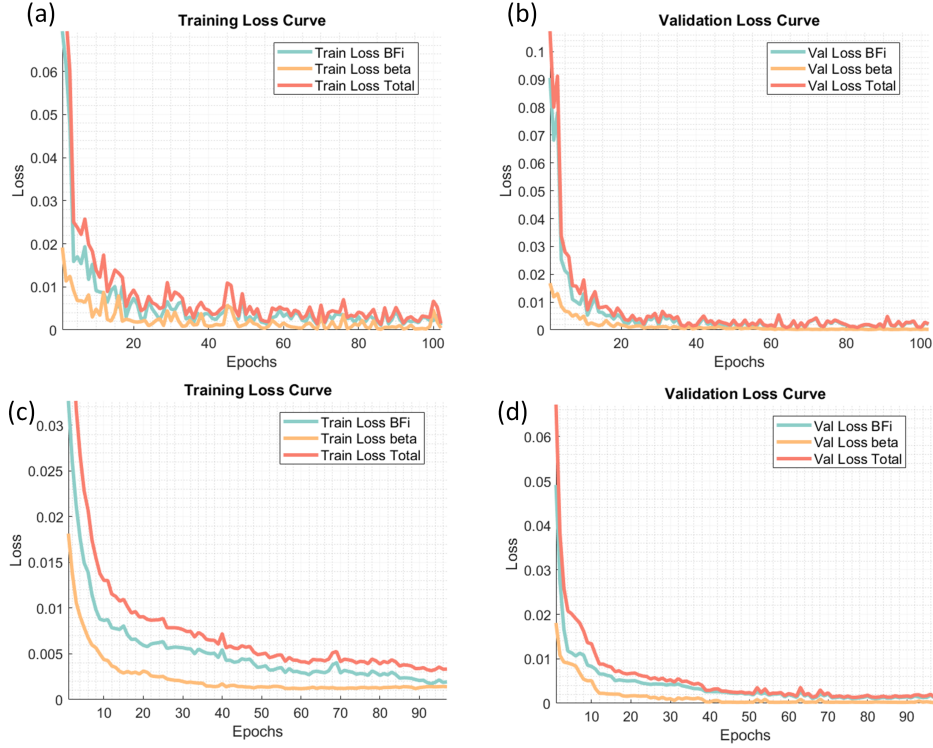


Figure 5.5: Training and validation loss curves of ACNN and CNN. (a) and (b), ACNN training and validation loss curves in 109 epochs. (c) and (d), CNN training and validation loss curves in 96 epochs.

Table 5.1: Detailed computational information of each layer.

	UAC1	BN1	UAC2	BN2	UAC3_1	BN3_1	UAC3_2	BN3_2	UAC3_3	BN3_3	Total
#parameters	270	30	2,940	30	2,730x2	60x2	465x2	30x2	16x2	2x2	9,880
FLOPs	10,260	1,140	17,640	180	2,730x2	60x2	465x2	30x2	16x2	2x2	35,830

Note: Some parameters and FLOPs multiply by two because of the branched structure.

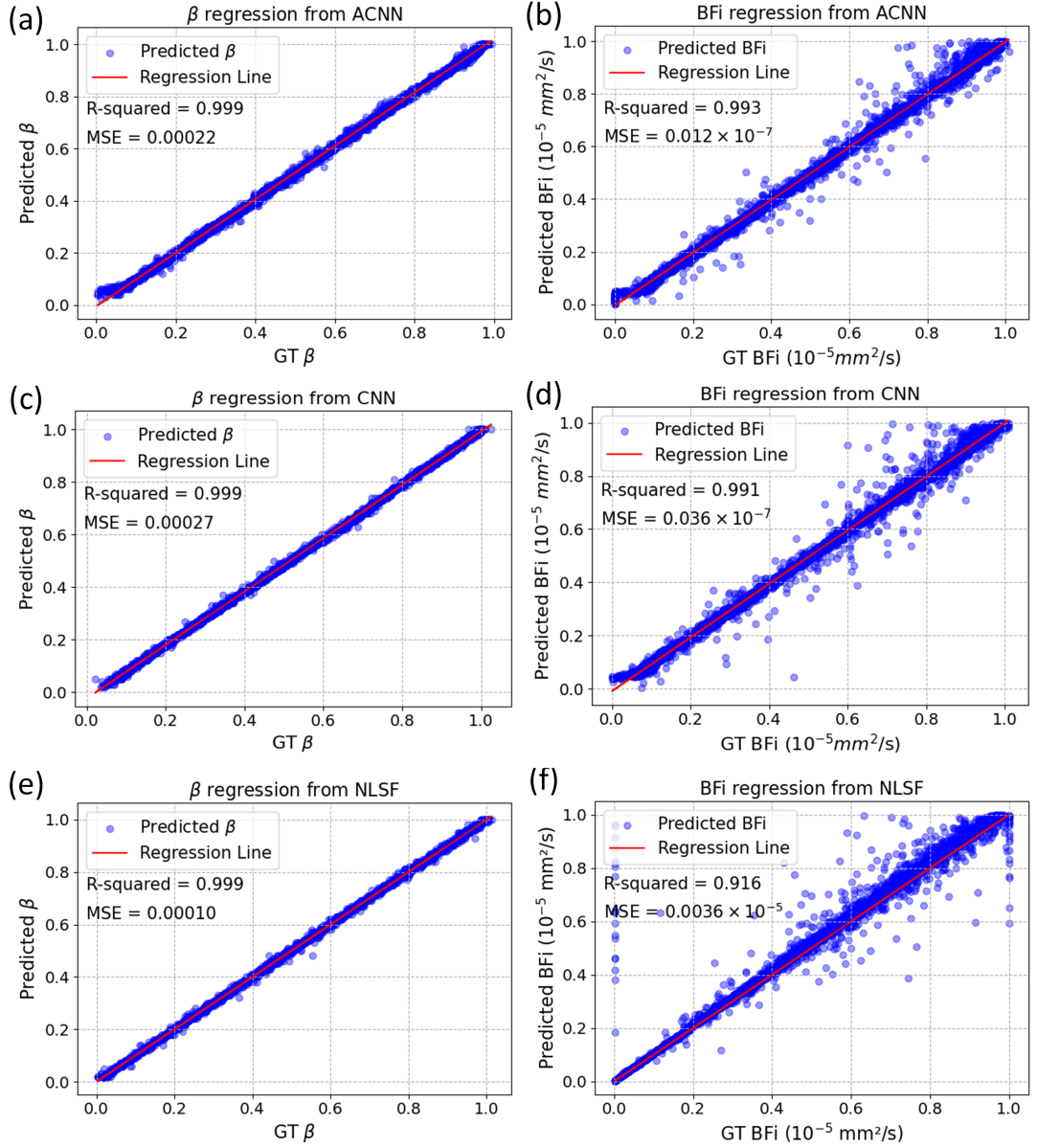


Figure 5.6: R -square and MSE evaluation between ACNN, CNN, NLSF. (a) and (b) R -square of BFi and from ACNN. (c) and (d) R -square of BFi and β from CNN. (e) and (f) R -square of BFi and β from NLSF.

the noise model and were normalized to emulate realistic $g_2(\tau)$. Figure 5.5, shows training and validation loss curves, where ACNN exhibits comparable convergence versus a CNN model with the same architecture and training strategy. The number of parameters and floating-point operations per second (FLOPs) for each layer was summarized in Table 5.1. The compact model size and low FLOPs pave the way for high-parallel hardware implementation afterwards.

5.4.1 Accuracy Evaluation

As our ACNN uses the l_1 norm to measure cross-correlation between weights and feature maps, we should ensure that ACNN's weights follow Laplace distributions according to [84]. As shown in Figure 5.7 (a) and (b), distribution weights from CNN and our ACNN exhibit Gaussian and Laplace distributions, in good agreement with the AdderNet theory [84]. To evaluate estimated BFi and β , in Figure 5.6, used R -squared and the MSE to evaluate the fitting goodness, in comparison with the GT BFi and β . Both ACNN and the conventional CNN offer nearly perfect R -squared results, However, for β , a few outliers from CNN (at small β) are shown. As for BFi's reconstruction, ACNN also attains better R -squared and MSE.

We also assessed the impact of the photon rate in the noise model on the reconstruction accuracy. We compared our ACNN with CNN at various photon intensity levels, ranging from 1,000 to 35,000 counts per second (CPS). This range was divided into 34 groups, each containing 100 noisy ACF curves. The GT β and BFi values are 0.5 and $0.5 \times 10^{-5} mm^2/s$, respectively. and t (the averaging time) = 5 seconds. As shown in Figure 5.8 (a), the accuracy of β (ACNN and CNN) is compromised when the photon rate is low, leading to inaccuracy in the mean (M) and the standard deviation (Std). However, as the photon intensity

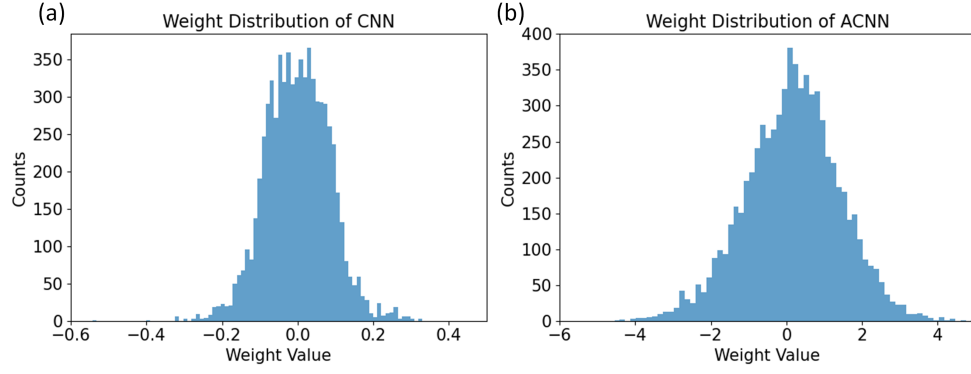


Figure 5.7: Histograms depict the weights distribution of CNN and ACNN. (a) and (b) represent Gaussian and Laplace distributions of CNN's and ACNN's weights.

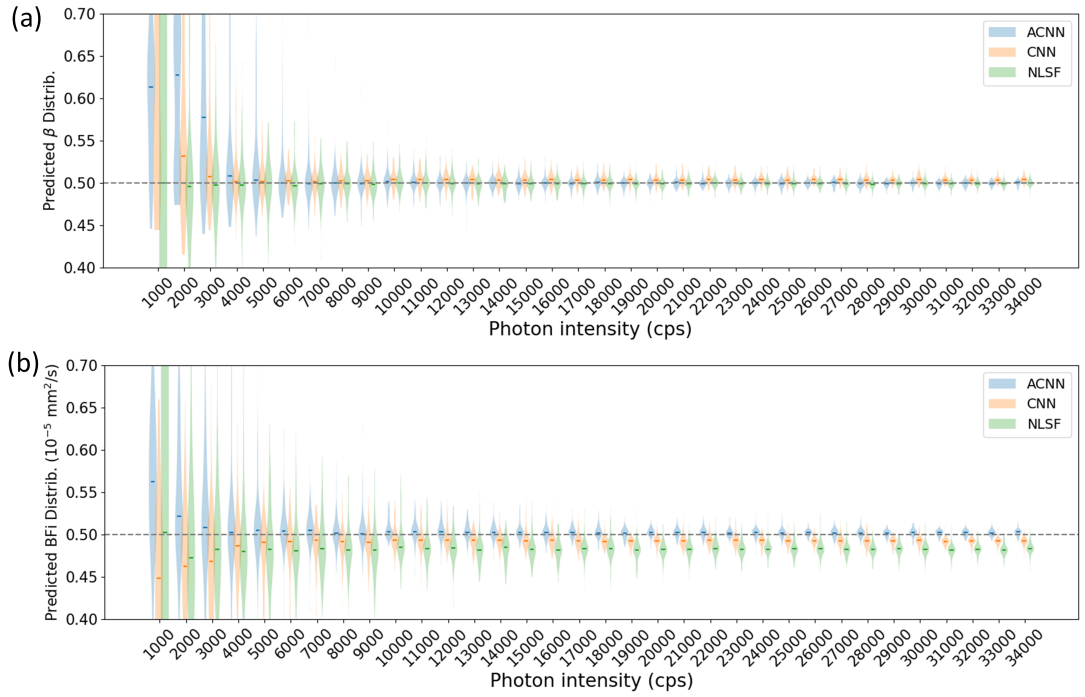


Figure 5.8: Accuracy evaluation of ACNN and CNN under different levels of photon rate in the noise model. (a) and (b) reconstructed β and BFi.

increases, both architectures provide more accurate reconstructions. ACNN tends to offer better accuracy for most photon intensities, whereas CNN exhibits slightly better accuracy (M) and precision (Std) when photon counts are lower than 9,000. Regarding BFi, ACNN outperforms CNN in terms of accuracy and achieves similar precision. Both architectures produce accurate BFi at high photon rates, whereas CNN is more robust for low photon rates (before 5,000 CPS).

5.4.2 Accelerator Architecture

The ACNN accelerator was implemented on PL on cost-optimized Zynq-7000 SoC and high-end Zynq-UltraScale+ MPSoC using Vivado HLS 2018.2. Alongside the accelerator, the computation of ACF was implemented on the ARM-based PS. The following reasons are for producing the ACF module on PS: firstly, implementing Eq.5.1 on the FPGA requires taking thousands of frames and conducting element-wise vector multiplications and divisions; it is challenging due to limited DSP slices and on-chip memory. The onboard DDR with hundreds of megabytes can efficiently accommodate raw data and bridge the data transfer between the ACF module on PS and BFi reconstruction on PL. Secondly, in theory, the multiplication operations in the numerator in Equation 5.1 are followed by an averaging operation that can be implemented as adder-trees and end up with a subsequent divider. Although the overhead of parallelizing the adder-tree for averaging is negligible, vector multiplications and divisions are computationally expensive in FPGAs. To address this concern, we leverage Neon and vector floating-point unit (VFPU) [139] technologies in ARM CPUs to accelerate vector multiplications in the numerator, as the arithmetic logical unit in ARM Cortex-A9 and -A53 CPU cores on our FPGA boards contain vectorized dividers and multipliers optimized with SIMD, which offloads ACF processing from PL.

We yielded 11-fold acceleration compared to the speed without enabling VFPU. Therefore, allocating workloads to PS and PL achieves a trade-off between performance and hardware utilization. Meanwhile, we can reserve more hardware resources for regular vector-addition and accumulation and VM operations for DL accelerators.

The processing pipeline in PS is summarized in Figure 5.9 (a). To validate the functionality, we initially preloaded intensity data in the DDR. Multiple bit-width FXP quantization methods were adopted to alleviate the hardware overhead and timing latency, and the FLP to FXP conversion was implemented as shift operations and divisions on PS. Once enough ACFs are computed and a pre-defined number corresponding to the number of cores specified in Table 5.2 is reached, the quantized and normalized ACF are dispatched to the PL accelerators through an AXI-full interface. Similarly, the FXP to FLP conversion was implemented similarly for printing readable BFi and β . *malloc*(\cdot) was used to store quantized ACF vectors and dispatch them to AXI’s Master port. *MACROS* parameterizes the pointer addressing space and the length of ACFs. A general-purpose timer driver was employed to measure the time-consumption when DL accelerators reconstruct BFi and β .

As shown in Figure 5.9, our ACNN accelerator is a scalable, multi-core architecture, where each DL-core can process one ACF and reconstruct BFi and β simultaneously. The details about each DL-core are depicted in Figure 5.9 (b); the input feature go through a series of UAC modules, where the for-loops of output channels and kernel windows are unrolled to improve the parallelism. The Reshape flattens the feature and allocates the data in channel dimensions for channel-wise convolution afterwards. Unrolling operations in UAC are decomposed in Figure 5.9 (c). The BRAMs (storing learned parameters) are partially

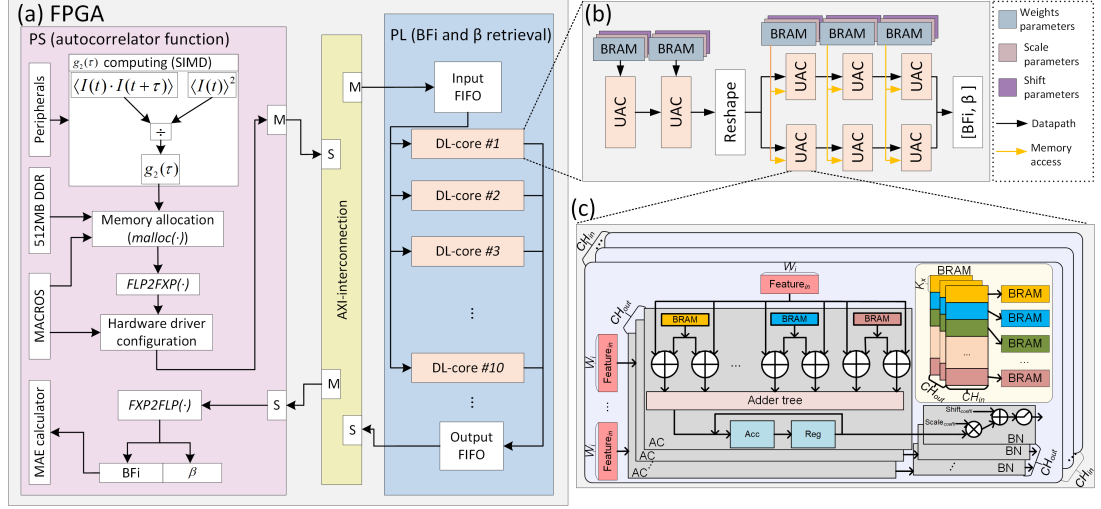


Figure 5.9: Hardware architecture integrates intensity temporal ACF computation and ACNN accelerators (10 cores on Zynq and 15 on Zynq UltraScale+). (a). The architecture overview illustrates the data transfer and functionalities of each module. (b). Detailed architecture of each DL core, data path and memory access were depicted with back and yellow arrows, respectively. (c). Detailed structures of each UAC illustrate parallelism in the input channel, output channel, and kernel size.

Table 5.2: Evaluation results of ACNN accelerators with different quantization bit-width on Zynq-7000 and Zynq-UltraScale+ FPGA.

FPGA	# Core	Data type	DFF	LUT	LUTRAM	BRAM_18K	DSP	PPMS
Zynq-7000	5	Fixed<18, 9>	18.99% (20,203/106,400)	50.56% (26,898/53,200)	7.20% (1,253/17,400)	36.07% (50.50/140)	29.55% (65/220)	21.74
	5	Fixed<24, 12>	19.00% (20,212/106,400)	49.89% (26,541/53,200)	7.20% (1,253/17,400)	36.07% (50.50/140)	29.55% (65/220)	17.86
	10	Fixed<18, 9>	34.75% (36,973/106,400)	91.93% (48,588/53,200)	12.55% (2,183/17,400)	75.36% (105.5/140)	59.09% (130/220)	41.67
	10	Fixed<24, 12>	34.81% (37,043/106,400)	90.07% (47,919/53,200)	12.54% (2,182/17,400)	75.36% (105.5/140)	59.09% (130/220)	35.01
Zynq UltraScale+ MPSoC	10	Fixed<18, 9>	5.88% (27,083/460,800)	21.96% (50,603/230,400)	1.35% (1,375/101,760)	35.58% (111/312)	7.52% (130/1,728)	38.46
	10	Fixed<24, 12>	5.68% (26,162/460,800)	22.25% (50,603/230,400)	1.35% (1,375/101,760)	35.58% (111/312)	7.52% (130/1,728)	33.21
	15	Fixed<18, 9>	8.35% (38,473/460,800)	32.28% (74,375/230,400)	1.82% (1,855/101,760)	53.21% (166/312)	11.28% (195/1,728)	48.49
	15	Fixed<24, 12>	8.05% (37,101/460,800)	32.74% (75,438/230,400)	1.82% (1,855/101,760)	53.21% (166/312)	11.28% (195/1,728)	44.37

Note: Some results may vary depending on the quantization bit-width and FPGA configuration.

partitioned accordingly to satisfy the bandwidth requirement of UACs. As BNs are involved in each UAC, they are implemented as VMs after ACs. BNs were simplified using Eq. 3.11, Eq. 3.12, and Eq. 3.13. Once these parameters from the pre-trained model are extracted, they are preloaded on BRAMs, as the dashed box shows in Figure 5.9 (b).

Table 5.3: Performance comparisons of CPU, GPU, and FPGA-based SoC when processing different numbers of pixels (ACFs) for each batch.

	5 pixels/batch			
	i7-12800H CPU	RTX A1000 GPU	Artix-7 and 1 thread ARM-Cortex-A9 <18, 9>	Artix-7 and 1 thread ARM-Cortex-A9 <24, 12>
Power (W)	3.586	4.438	1.962	1.981
Latency (ms/batch)	3.753	1.311	0.183	0.174
Efficiency (ms/W)	1.045	0.295	0.095	0.015
	10 pixels/batch			
	i7-12800H CPU	RTX A1000 GPU	Artix-7 and 1 thread ARM-Cortex-A9 <18, 9>	Artix-7 and 1 thread ARM-Cortex-A9 <24, 12>
Power (W)	3.534	4.346	2.549	2.579
Latency (ms/batch)	3.927	1.356	0.258	0.267
Efficiency (ms/W)	1.111	0.312	0.101	0.104
	15 pixels/batch			
	i7-12800H CPU	RTX A1000 GPU	UltraScale+ and 1 thread ARM-Cortex-A53 <18, 9>	UltraScale+ and 1 thread ARM-Cortex-A53 <24, 12>
Power (W)	3.549	4.236	4.826	4.846
Latency (ms/batch)	5.115	1.524	0.283	0.295
Efficiency (ms/W)	1.441	0.360	0.058	0.061

5.4.3 Accelerator Evaluation

We implemented the ACNN accelerator on Zynq-7000 and Zynq-UltraScale+ to investigate its performance. To validate the scalability, we implemented five and ten DL cores on Zynq-7000, and ten and fifteen on Zynq UltraScale+. Each scheme contains two sets of FXP bit-width, 24 bit-width containing 12 fractional bits, and 18 bit-width containing 9 fractional bits. Hardware consumption and speed are demonstrated in Table 5.2. Considering our accelerator targets single-

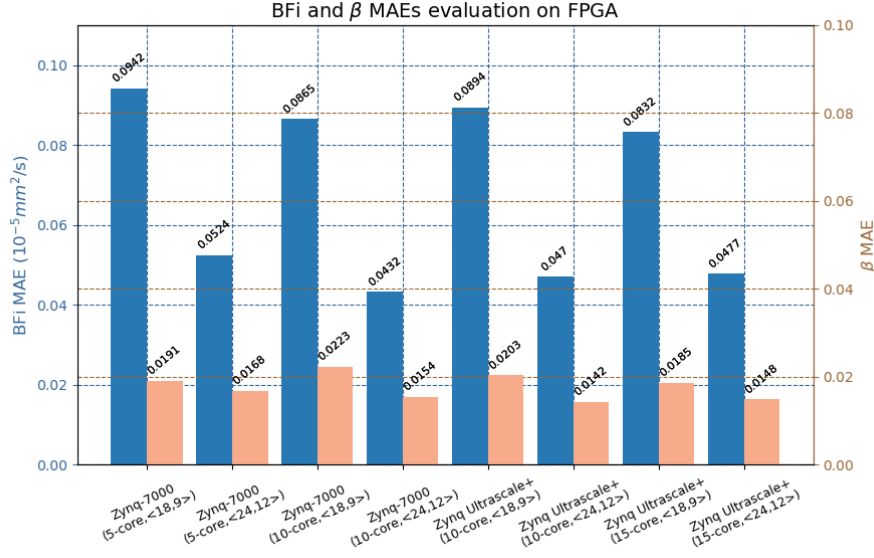


Figure 5.10: DL cores implementation on Zynq-7000 and Zynq-UltraScale+ with different quantization FXP schemes, and their corresponding MAE of BFI and β .

photon detector arrays, we use PPMS to assess the speed of simultaneously processing ACFs. Timer functions on the PS measure the time-consumption of the ACNN accelerator. According to Table 5.2, PPMS increases while the number of DL cores increases. Besides, we evaluated the reconstruction accuracy of BFI and β of the eight schemes in terms of MAE that is directly output from the FPGA board. As shown in Figure 5.10, there is no considerable fluctuation from different numbers of cores and FXP bit-widths. We define the computational efficiency using the latency of batches over power consumption, as shown in Table 5.3. It measures the time required to complete a given workload under constant power consumption. Therefore, a lower value indicates higher computational efficiency. We measured the power consumption of accelerators on FPGAs using Xilinx Power Estimator [140]. We also used NVML [139] and PyJoules [141]

APIs to measure the power consumption of GPU and CPU. FPGA-based SoC platforms obtain the best computational efficiency across different batch sizes compared to the CPU and GPU.

5.5 Summary and Future Work

This study reports a compact, hardware friendly DNN to reconstruct BFi and β in DCS. The correctness of our analytical model was verified by comparing *in-silico* MC simulations. According to Fig. 5.6, we ensure that our DNN architecture achieves higher accuracy than conventional CNNs yet has simpler operators. By leveraging the miniaturized architecture, we implemented our scalable DL clusters on an SoC-based FPGA to assess their accuracy, hardware utilization, and computational efficiency. Enhanced with different bit-widths of quantization schemes, we evaluated these quantized versions on cost-optimized and high-end FPGAs. To achieve end-to-end processing, we encapsulated $g_2(\tau)$ generation and BFi reconstruction on a single SoC-based FPGA, providing different on-chip processing solutions for modern DCS sensing systems. Given that current FPGAs mounted on modern SPAD arrays are mainly middle-end devices that only contain PL, our heterogenous hardware architecture provides a foundation for future integration of high-end FPGA and SPAD arrays.

Chapter 6

Conclusions and Final Remarks

6.1 Summary of Remarks

In this thesis, we illustrate the effectiveness of data-driven DL strategies in bolstering both accuracy and computational efficiency in parameter retrieval across three burgeoning applications within single-photon signal processing: depth reconstruction for single-photon LiDAR, FLIM analysis, and BFi reconstruction for DCS.

The comprehensive technical conclusions for each application are provided at the conclusion of their respective chapters. A unified methodology is applied in each application. We harness analytical models of optical instruments and single-photon data encoding to generate synthetic data for training DNNs. Furthermore, we delve into an investigation on how the DNNs' accuracy is affected by SNR, with results indicating that our DNNs, whether employing compression schemes or not, surpass conventional iterative optimization-based algorithms. Emphasizing the importance of data compression or DNN model optimization, we underscore their pivotal role in expediting parameter estimation, rather than solely concentrating

on algorithmic development. We develop hardware-optimized DNNs on FPGA to underscore their computational efficiency. Surpassing the performance of traditional CPU and GPU setups suggests a promising avenue towards embedded computing solutions, supplanting the need for post-processing methodologies.

Overall, our work demonstrates the effectiveness of DNNs for analyzing encoded single-photon signals, specifically through the generation of histograms for LiDAR and FLIM, and ACFs for DCS. We anticipate that the analytical models, streamlined DNN construction, compression techniques, and hardware architecture discussed in this study will contribute to improving parameter reconstruction accuracy and computational efficiency within the respective communities.

6.2 Potential Future Works

In the realm of single-photon LiDAR, achieving real-time and accurate depth reconstruction poses ongoing challenges. While a recent study [142] successfully utilized sparsity within 3D point-cloud data from SPAD technology to attain video-rate 3D reconstruction, a significant limitation persists. This approach relied on synthetic training data tailored specifically for bespoke SPAD sensors, meticulously considering optical parameters such as IRF, quantum efficiency, and TDC temporal resolution. However, the performance of such algorithms may degrade for different optical setups. To address this challenge, there is a critical need to develop robust algorithms capable of leveraging data sparsity while demonstrating adaptability across diverse optical configurations with minimal adjustments required.

In FLIM analysis, constructing histograms not only requires additional on-chip logic in the sensor or firmware on FPGA but also introduces latency for lifetime reconstruction since histograms need to be available in advance. Recent studies [81], [143] demonstrate that the raw data of photon-arrival timestamps can be directly input into a recurrent neural network and a spiking neural network without constructing histograms, thus mitigating latency concerns. This approach offers a promising direction for future research. Firstly, validating the robustness of timestamps-based methods under low-photon count scenarios is essential. Secondly, efforts should focus on enabling timestamps-based methods to reconstruct individual and average fluorescence lifetime effectively.

In our current approach to DCS, we use a single-layer semi-infinite analytical model to generate synthetic data for training our DNN. However, human tissues, like the brain, have a complex multi-layer structure. Moving forward, it's crucial to develop an accurate multi-layer analytical model that can effectively capture this complexity. Validating this model through MC simulation techniques will ensure its ability to accurately represent the intricate behavior of diffusion model within multi-layered tissues. Another promising track for future work is integrating our SoC-based FPGA accelerator into DCS systems. In contemporary SPAD-based DCS systems, FPGAs typically offer mid-level performance, featuring pure PL fabric without high-performance hardcore processors. For example, Artix-7 [6] and Kintex-7 [110] have been used to compute $g_2(\tau)$ directly on the FPGA without performing BFi reconstruction following $g_2(\tau)$. In contrast, we utilized the powerful hardcore ARM CPU cores with SIMD acceleration on the UltraScale+ MPSoC to compute $g_2(\tau)$ and generate BFi. However, existing FPGAs like Artix-7 and Kintex-7 do not include hardcore ARM CPUs. Although synthesizable soft-core processors in the PL fabric are an option, they are less

performant than hardcore processors and consume significant hardware resources. Therefore, we anticipate that our holistic accelerator prototype could significantly enhance system capabilities once high-end FPGAs like the UltraScale+ MPSoC are integrated into DCS systems.

This thesis introduces data-driven single-photon signal processing frameworks for LiDAR, FLIM, and DCS, laying the groundwork for future research. Besides the applications discussed, we aim to make our algorithms and hardware implementations adaptable to other fields utilizing single-photon detectors, such as fluorescence diffuse spectroscopy and diffuse optical tomography.

References

- [1] J. Rapp, J. Tachella, Y. Altmann, S. McLaughlin, and V. K. Goyal, “Advances in single-photon lidar for autonomous vehicles: Working principles, challenges, and recent advances,” *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 62–71, 2020.
- [2] J. Tachella, Y. Altmann, N. Mellado, A. McCarthy, R. Tobin, G. S. Buller, J.-Y. Tournet, and S. McLaughlin, “Real-time 3d reconstruction from single-photon lidar data using plug-and-play point cloud denoisers,” *Nature communications*, vol. 10, no. 1, p. 4984, 2019.
- [3] D. D.-U. Li, J. Arlt, D. Tyndall, R. Walker, J. Richardson, D. Stoppa, E. Charbon, and R. K. Henderson, “Video-rate fluorescence lifetime imaging camera with cmos single-photon avalanche diode arrays and high-speed imaging algorithm,” *Journal of biomedical optics*, vol. 16, no. 9, pp. 096012–096012, 2011.
- [4] V. Zickus, M.-L. Wu, K. Morimoto, V. Kapitany, A. Fatima, A. Turpin, R. Insall, J. Whitelaw, L. Machesky, C. Bruschini, *et al.*, “Fluorescence lifetime imaging with a megapixel spad camera and neural network lifetime estimation,” *Scientific Reports*, vol. 10, no. 1, p. 20986, 2020.

REFERENCES

- [5] W. Liu, R. Qian, S. Xu, P. Chandra Konda, J. Jönsson, M. Harfouche, D. Borycki, C. Cooke, E. Berrocal, Q. Dai, *et al.*, “Fast and sensitive diffuse correlation spectroscopy with highly parallelized single photon detection,” *APL Photonics*, vol. 6, no. 2, 2021.
- [6] F. M. Della Rocca, E. J. Sie, R. Catoen, F. Marsili, and R. K. Henderson, “Field programmable gate array compression for large array multispeckle diffuse correlation spectroscopy,” *Journal of Biomedical Optics*, vol. 28, no. 5, p. 057001, 2023.
- [7] D. Shin, A. Kirmani, V. K. Goyal, and J. H. Shapiro, “Photon-efficient computational 3-d and reflectivity imaging with single-photon detectors,” *IEEE Transactions on Computational Imaging*, vol. 1, no. 2, pp. 112–125, 2015.
- [8] J. Rapp and V. K. Goyal, “A few photons among many: Unmixing signal and noise for photon-efficient active imaging,” *IEEE Transactions on Computational Imaging*, vol. 3, no. 3, pp. 445–459, 2017.
- [9] J. Lee, J. V. Chacko, B. Dai, S. A. Reza, A. K. Sagar, K. W. Eliceiri, A. Velten, and M. Gupta, “Coding scheme optimization for fast fluorescence lifetime imaging,” *ACM Transactions on Graphics (TOG)*, vol. 38, no. 3, pp. 1–16, 2019.
- [10] Y. Zhang, Y. Chen, and D. D.-U. Li, “Optimizing laguerre expansion based deconvolution methods for analysing bi-exponential fluorescence lifetime images,” *Optics express*, vol. 24, no. 13, pp. 13894–13905, 2016.

REFERENCES

- [11] L. Dong, L. He, Y. Lin, Y. Shang, and G. Yu, “Simultaneously extracting multiple parameters via fitting one single autocorrelation function curve in diffuse correlation spectroscopy,” *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 2, pp. 361–368, 2012.
- [12] M. Beer, O. M. Schrey, J. F. Haase, J. Ruskowski, W. Brockherde, B. J. Hosticka, and R. Kokozinski, “Spad-based flash lidar sensor with high ambient light rejection for automotive applications,” in *Quantum Sensing and Nano Electronics and Photonics XV*, vol. 10540, pp. 320–327, SPIE, 2018.
- [13] L. Nalpantidis and A. Gasteratos, “Stereo vision depth estimation methods for robotic applications,” in *Depth Map and 3D Imaging Applications: Algorithms and Technologies*, pp. 397–417, IGI global, 2012.
- [14] A. N. Angelopoulos, H. Ameri, D. Mitra, and M. Humayun, “Enhanced depth navigation through augmented reality depth mapping in patients with low vision,” *Scientific reports*, vol. 9, no. 1, p. 11230, 2019.
- [15] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, “Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 340–349, 2018.
- [16] Y. Wang, Z. Lai, G. Huang, B. H. Wang, L. Van Der Maaten, M. Campbell, and K. Q. Weinberger, “Anytime stereo image depth estimation on mobile devices,” in *2019 international conference on robotics and automation (ICRA)*, pp. 5893–5900, IEEE, 2019.

REFERENCES

- [17] E. Cippitelli, F. Fioranelli, E. Gambi, and S. Spinsante, “Radar and rgb-depth sensors for fall detection: A review,” *IEEE Sensors Journal*, vol. 17, no. 12, pp. 3585–3604, 2017.
- [18] C. Niclass, A. Rochas, P.-A. Besse, and E. Charbon, “Design and characterization of a cmos 3-d image sensor based on single photon avalanche diodes,” *IEEE Journal of Solid-State Circuits*, vol. 40, no. 9, pp. 1847–1854, 2005.
- [19] J. A. Richardson, L. A. Grant, and R. K. Henderson, “Low dark count single-photon avalanche diode structure compatible with standard nanometer scale cmos technology,” *IEEE Photonics Technology Letters*, vol. 21, no. 14, pp. 1020–1022, 2009.
- [20] C. Veerappan, J. Richardson, R. Walker, D.-U. Li, M. W. Fishburn, Y. Maruyama, D. Stoppa, F. Borghetti, M. Gersbach, R. K. Henderson, *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*, pp. 312–314, IEEE, 2011.
- [21] R. K. Henderson, N. Johnston, F. Mattioli Della Rocca, H. Chen, D. Day-Uei Li, G. Hungerford, R. Hirsch, D. Mcloskey, P. Yip, and D. J. S. Birch, “A 192×128 time correlated spad image sensor in 40-nm cmos technology,” *IEEE Journal of Solid-State Circuits*, vol. 54, no. 7, pp. 1907–1916, 2019.

REFERENCES

- [22] Z. T. Harmany, R. F. Marcia, and R. M. Willett, “This is spiral-tap: Sparse poisson intensity reconstruction algorithms—theory and practice,” *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1084–1096, 2011.
- [23] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.
- [24] J. F. C. Kingman, *Poisson processes*, vol. 3. Clarendon Press, 1992.
- [25] J. Tachella, Y. Altmann, X. Ren, A. McCarthy, G. S. Buller, S. McLaughlin, and J.-Y. Tournet, “Bayesian 3d reconstruction of complex scenes from single-photon lidar data,” *SIAM Journal on Imaging Sciences*, vol. 12, no. 1, pp. 521–550, 2019.
- [26] Q. Legros, S. Meignen, S. McLaughlin, and Y. Altmann, “Expectation-maximization based approach to 3d reconstruction from single-waveform multispectral lidar data,” *IEEE Transactions on Computational Imaging*, vol. 6, pp. 1033–1043, 2020.
- [27] D. B. Lindell, M. O’Toole, and G. Wetzstein, “Single-photon 3d imaging with deep sensor fusion,” *ACM Trans. Graph.*, vol. 37, no. 4, pp. 113–1, 2018.
- [28] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical image computing and computer-assisted intervention—MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241, Springer, 2015.

REFERENCES

- [29] A. Ruget, S. McLaughlin, R. K. Henderson, I. Gyongy, A. Halimi, and J. Leach, “Robust super-resolution depth imaging via a multi-feature fusion deep network,” *Optics Express*, vol. 29, no. 8, pp. 11917–11937, 2021.
- [30] Z. Sun, D. B. Lindell, O. Solgaard, and G. Wetzstein, “Spadnet: deep rgb-spada sensor fusion assisted by monocular depth estimation,” *Optics express*, vol. 28, no. 10, pp. 14948–14962, 2020.
- [31] J. Peng, Z. Xiong, X. Huang, Z.-P. Li, D. Liu, and F. Xu, “Photon-efficient 3d imaging with a non-local neural network,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pp. 225–241, Springer, 2020.
- [32] X. Zhao, M. Wu, Y. Zhang, C. Wang, R. Chen, W. He, and Q. Chen, “Robust single-photon 3d imaging based on full-scale feature integration and intensity edge guidance,” *Optics and Lasers in Engineering*, vol. 172, p. 107850, 2024.
- [33] J. Arlt, D. Tyndall, B. R. Rae, D. D.-U. Li, J. A. Richardson, and R. K. Henderson, “A study of pile-up in integrated time-correlated single photon counting systems,” *Review of Scientific Instruments*, vol. 84, no. 10, 2013.
- [34] D. L. Snyder and M. I. Miller, *Random point processes in time and space*. Springer Science & Business Media, 2012.
- [35] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, “Unet++: Redesigning skip connections to exploit multiscale features in image segmentation,” *IEEE transactions on medical imaging*, vol. 39, no. 6, pp. 1856–1867, 2019.

REFERENCES

- [36] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, “Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients,” *arXiv preprint arXiv:1606.06160*, 2016.
- [37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [38] J. Liu, Y. Sun, X. Xu, and U. S. Kamilov, “Image restoration using total variation regularized deep image prior,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7715–7719, Ieee, 2019.
- [39] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from rgb-d images,” in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pp. 746–760, Springer, 2012.
- [40] D. Scharstein and C. Pal, “Learning conditional random fields for stereo,” in *2007 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2007.
- [41] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [42] S. Burri, C. Bruschini, and E. Charbon, “Linospad: a compact linear spad camera system with 64 fpga-based tdc modules for versatile 50 ps resolution time-resolved imaging,” *Instruments*, vol. 1, no. 1, p. 6, 2017.
- [43] I. Alhashim and P. Wonka, “High quality monocular depth estimation via transfer learning. arxiv 2018,” *arXiv preprint arXiv:1812.11941*, 1812.

REFERENCES

- [44] I. Bar-David, “Communication under the poisson regime,” *IEEE Transactions on Information Theory*, vol. 15, no. 1, pp. 31–37, 1969.
- [45] M. A. Yaseen, S. Sakadžić, W. Wu, W. Becker, K. A. Kasischke, and D. A. Boas, “In vivo imaging of cerebral energy metabolism with two-photon fluorescence lifetime microscopy of nadh,” *Biomedical optics express*, vol. 4, no. 2, pp. 307–321, 2013.
- [46] Y. Dancik, A. Favre, C. J. Loy, A. V. Zvyagin, and M. S. Roberts, “Use of multiphoton tomography and fluorescence lifetime imaging to investigate skin pigmentation in vivo,” *Journal of biomedical optics*, vol. 18, no. 2, pp. 026022–026022, 2013.
- [47] M. Nobis, E. J. McGhee, J. P. Morton, J. P. Schwarz, S. A. Karim, J. Quinn, M. Edward, A. D. Campbell, L. C. McGarry, T. J. Evans, *et al.*, “Intravital flim-fret imaging reveals dasatinib-induced spatial control of src in pancreatic cancer,” *Cancer research*, vol. 73, no. 15, pp. 4674–4686, 2013.
- [48] S. Karpf, C. T. Riche, D. Di Carlo, A. Goel, W. A. Zeiger, A. Suresh, C. Portera-Cailliau, and B. Jalali, “Spectro-temporal encoded multiphoton microscopy and fluorescence lifetime imaging at kilohertz frame-rates,” *Nature communications*, vol. 11, no. 1, p. 2062, 2020.
- [49] Y. Sun, N. Hatami, M. Yee, J. Phipps, D. S. Elson, F. Gorin, R. J. Schrot, and L. Marcu, “Fluorescence lifetime imaging microscopy for brain tumor image-guided surgery,” *Journal of biomedical optics*, vol. 15, no. 5, pp. 056022–056022, 2010.

REFERENCES

- [50] A. Alfonso-Garcia, J. Bec, S. Sridharan Weaver, B. Hartl, J. Unger, M. Bobinski, M. Lechpammer, F. Girgis, J. Boggan, and L. Marcu, “Real-time augmented reality for delineation of surgical margins during neurosurgery using autofluorescence lifetime contrast,” *Journal of biophotonics*, vol. 13, no. 1, p. e201900108, 2020.
- [51] W. Becker, A. Bergmann, G. L. Biscotti, and A. Rueck, “Advanced time-correlated single photon counting techniques for spectroscopy and imaging in biomedical systems,” in *Commercial and Biomedical Applications of Ultrafast Lasers IV*, vol. 5340, pp. 104–112, SPIE, 2004.
- [52] K. Suhling, L. M. Hirvonen, W. Becker, S. Smietana, H. Netz, J. Milnes, T. Conneely, A. Le Marois, O. Jagutzki, F. Festy, *et al.*, “Wide-field time-correlated single photon counting-based fluorescence lifetime imaging microscopy,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 942, p. 162365, 2019.
- [53] J. C. Thiele, D. A. Helmerich, N. Oleksiievets, R. Tsukanov, E. Butkevich, M. Sauer, O. Nevskiy, and J. Enderlein, “Confocal fluorescence-lifetime single-molecule localization microscopy,” *ACS nano*, vol. 14, no. 10, pp. 14190–14200, 2020.
- [54] J. Philip and K. Carlsson, “Theoretical investigation of the signal-to-noise ratio in fluorescence lifetime imaging,” *JOSA A*, vol. 20, no. 2, pp. 368–379, 2003.
- [55] J. R. Lakowicz, *Principles of fluorescence spectroscopy*. Springer, 2006.

REFERENCES

- [56] Ž. Bajzer and F. G. Prendergast, “[10] maximum likelihood analysis of fluorescence data,” in *Methods in enzymology*, vol. 210, pp. 200–237, Elsevier, 1992.
- [57] D. D.-U. Li, H. Yu, and Y. Chen, “Fast bi-exponential fluorescence lifetime imaging analysis methods,” *Optics Letters*, vol. 40, no. 3, pp. 336–339, 2015.
- [58] A. Sillen and Y. Engelborghs, “The correct use of “average” fluorescence parameters,” *Photochemistry and photobiology*, vol. 67, no. 5, pp. 475–486, 1998.
- [59] Y. Li, S. Natakorn, Y. Chen, M. Safar, M. Cunningham, J. Tian, and D. D.-U. Li, “Investigations on average fluorescence lifetimes for visualizing multi-exponential decays,” *Frontiers in physics*, vol. 8, p. 576862, 2020.
- [60] G. Wu, T. Nowotny, Y. Zhang, H.-Q. Yu, and D. D.-U. Li, “Artificial neural network approaches for fluorescence lifetime imaging techniques,” *Optics letters*, vol. 41, no. 11, pp. 2561–2564, 2016.
- [61] Q. Wang, Y. Li, D. Xiao, Z. Zang, Z. Jiao, Y. Chen, and D. D. U. Li, “Simple and robust deep learning approach for fast fluorescence lifetime imaging,” *Sensors*, vol. 22, no. 19, p. 7293, 2022.
- [62] D. Xiao, Y. Chen, and D. D.-U. Li, “One-dimensional deep learning architecture for fast fluorescence lifetime imaging,” *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 27, no. 4, pp. 1–10, 2021.

REFERENCES

- [63] L. Héliot and A. Leray, “Simple phasor-based deep neural network for fluorescence lifetime imaging microscopy,” *Scientific Reports*, vol. 11, no. 1, p. 23858, 2021.
- [64] J. T. Smith, R. Yao, N. Sinsuebphon, A. Rudkouskaya, N. Un, J. Mazurkiewicz, M. Barroso, P. Yan, and X. Intes, “Fast fit-free analysis of fluorescence lifetime imaging via deep learning,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 48, pp. 24019–24030, 2019.
- [65] M. Ochoa, A. Rudkouskaya, R. Yao, P. Yan, M. Barroso, and X. Intes, “High compression deep learning based single-pixel hyperspectral macroscopic fluorescence lifetime imaging in vivo,” *Biomedical Optics Express*, vol. 11, no. 10, pp. 5401–5424, 2020.
- [66] Y.-I. Chen, Y.-J. Chang, S.-C. Liao, T. D. Nguyen, J. Yang, Y.-A. Kuo, S. Hong, Y.-L. Liu, H. G. Rylander III, S. R. Santacruz, *et al.*, “Generative adversarial network enables rapid and robust fluorescence lifetime image analysis in live cells,” *Communications Biology*, vol. 5, no. 1, p. 18, 2022.
- [67] J. T. Smith, M. Ochoa, and X. Intes, “Unmix-me: spectral and lifetime fluorescence unmixing via deep learning,” *Biomedical Optics Express*, vol. 11, no. 7, pp. 3857–3874, 2020.
- [68] D. Xiao, Z. Zang, W. Xie, N. Sapermsap, Y. Chen, and D. D. U. Li, “Spatial resolution improved fluorescence lifetime imaging via deep learning,” *Optics Express*, vol. 30, no. 7, pp. 11479–11494, 2022.
- [69] Z. Zang, D. Xiao, Q. Wang, Z. Li, W. Xie, Y. Chen, and D. D. U. Li, “Fast analysis of time-domain fluorescence lifetime imaging via extreme learning machine,” *Sensors*, vol. 22, no. 10, p. 3758, 2022.

REFERENCES

- [70] D. Xiao, N. Sapermsap, Y. Chen, and D. D. U. Li, “Deep learning enhanced fast fluorescence lifetime imaging with a few photons,” *Optica*, vol. 10, pp. 944–951, Jul 2023.
- [71] B. Kim, B. Park, S. Lee, and Y. Won, “Gpu accelerated real-time confocal fluorescence lifetime imaging microscopy (flim) based on the analog mean-delay (amd) method,” *Biomedical Optics Express*, vol. 7, no. 12, pp. 5055–5065, 2016.
- [72] J. E. Sorrells, R. R. Iyer, L. Yang, A. J. Bower, D. R. Spillman, E. J. Chaney, H. Tu, and S. A. Boppart, “Real-time pixelwise phasor analysis for video-rate two-photon fluorescence lifetime imaging microscopy,” *Biomedical Optics Express*, vol. 12, no. 7, pp. 4003–4019, 2021.
- [73] Y. Ma, Y. Lee, C. Best-Popescu, and L. Gao, “High-speed compressed-sensing fluorescence lifetime imaging microscopy of live cells,” *Proceedings of the National Academy of Sciences*, vol. 118, no. 3, p. e2004176118, 2021.
- [74] I. Gyongy, N. Calder, A. Davies, N. A. W. Dutton, R. R. Duncan, C. Rickman, P. Dalgarno, and R. K. Henderson, “A 256×256 , 100-kfps, 61fill-factor spad image sensor for time-resolved microscopy applications,” *IEEE Transactions on Electron Devices*, vol. 65, no. 2, pp. 547–554, 2018.
- [75] D.-U. Li, B. Rae, R. Andrews, J. Arlt, and R. Henderson, “Hardware implementation algorithm and error analysis of high-speed fluorescence lifetime sensing systems using center-of-mass method,” *Journal of biomedical optics*, vol. 15, no. 1, pp. 017006–017006, 2010.

REFERENCES

- [76] D. Tyndall, B. R. Rae, D. D.-U. Li, J. Arlt, A. Johnston, J. A. Richardson, and R. K. Henderson, “A high-throughput time-resolved mini-silicon photomultiplier with embedded fluorescence lifetime estimation in 0.13 μm cmos,” *IEEE transactions on biomedical circuits and systems*, vol. 6, no. 6, pp. 562–570, 2012.
- [77] N. Krstajić, J. Levitt, S. Poland, S. Ameer-Beg, and R. Henderson, “256 \times 2 spad line sensor for time resolved fluorescence spectroscopy,” *Optics express*, vol. 23, no. 5, pp. 5653–5669, 2015.
- [78] F. M. Della Rocca, J. Nedbal, D. Tyndall, N. Krstajić, D. D.-U. Li, S. M. Ameer-Beg, and R. K. Henderson, “Real-time fluorescence lifetime actuation for cell sorting using a cmos spad silicon photomultiplier,” *Optics letters*, vol. 41, no. 4, pp. 673–676, 2016.
- [79] D. Xiao, Z. Zang, N. Sapermsap, Q. Wang, W. Xie, Y. Chen, and D. D. U. Li, “Dynamic fluorescence lifetime sensing with cmos single-photon avalanche diode arrays and deep learning processors,” *Biomed. Opt. Express*, vol. 12, pp. 3450–3462, Jun 2021.
- [80] M. J. Serafino, B. E. Applegate, and J. A. Jo, “Direct frequency domain fluorescence lifetime imaging using field programmable gate arrays for real time processing,” *Review of Scientific Instruments*, vol. 91, no. 3, 2020.
- [81] Y. Lin, P. Mos, A. Ardelean, C. Bruschini, and E. Charbon, “Coupling a recurrent neural network to spad tcspc systems for real-time fluorescence lifetime imaging,” *Scientific Reports*, vol. 14, no. 1, pp. 1–13, 2024.

REFERENCES

- [82] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, “Binarized neural networks,” *Advances in neural information processing systems*, vol. 29, 2016.
- [83] Y. Wang, M. Huang, K. Han, H. Chen, W. Zhang, C. Xu, and D. Tao, “Addernet and its minimalist hardware design for energy-efficient artificial intelligence,” *arXiv preprint arXiv:2101.10015*, 2021.
- [84] H. Chen, Y. Wang, C. Xu, B. Shi, C. Xu, Q. Tian, and C. Xu, “Addernet: Do we really need multiplications in deep learning?,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1468–1477, 2020.
- [85] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [86] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, “A survey of fpga-based neural network inference accelerator (2018),” *arXiv preprint arXiv:1712.08934*.
- [87] A. L. Trinh and A. Esposito, “Biochemical resolving power of fluorescence lifetime imaging: untangling the roles of the instrument response function and photon-statistics,” *Biomedical Optics Express*, vol. 12, no. 7, pp. 3775–3788, 2021.
- [88] D. Kage, K. Hoffmann, G. Nifontova, V. Krivenkov, A. Sukhanova, I. Nabiev, and U. Resch-Genger, “Tempo-spectral multiplexing in flow cytometry with lifetime detection using qd-encoded polymer beads,” *Scientific Reports*, vol. 10, no. 1, p. 653, 2020.

REFERENCES

- [89] A. J. Bares, M. A. Mejjooli, M. A. Pender, S. A. Leddon, S. Tilley, K. Lin, J. Dong, M. Kim, D. J. Fowell, N. Nishimura, *et al.*, “Hyperspectral multiphoton microscopy for in vivo visualization of multiple, spectrally overlapped fluorescent labels,” *Optica*, vol. 7, no. 11, pp. 1587–1601, 2020.
- [90] M. A. Digman, V. R. Caiolfa, M. Zamai, and E. Gratton, “The phasor approach to fluorescence lifetime imaging analysis,” *Biophysical journal*, vol. 94, no. 2, pp. L14–L16, 2008.
- [91] D. Tamborini, K. A. Stephens, M. M. Wu, P. Farzam, A. M. Siegel, O. Shatrovoy, M. Blackwell, D. A. Boas, S. A. Carp, and M. A. Franceschini, “Portable system for time-domain diffuse correlation spectroscopy,” *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 11, pp. 3014–3025, 2019.
- [92] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.
- [93] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, “Extreme learning machine for regression and multiclass classification,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2011.
- [94] H.-T. Li, C.-Y. Chou, Y.-T. Chen, S.-H. Wang, and A.-Y. Wu, “Robust and lightweight ensemble extreme learning machine engine based on eigenspace domain for compressed learning,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 12, pp. 4699–4712, 2019.

REFERENCES

- [95] D. M. Jameson, E. Gratton, and R. D. Hall, “The measurement and analysis of heterogeneous emissions by multifrequency phase and modulation fluorometry,” *Applied spectroscopy reviews*, vol. 20, no. 1, pp. 55–106, 1984.
- [96] Z. Heger, M. Kominkova, N. Cernei, L. Krejcova, P. Kopel, O. Zitka, V. Adam, and R. Kizek, “Fluorescence resonance energy transfer between green fluorescent protein and doxorubicin enabled by dna nanotechnology,” *Electrophoresis*, vol. 35, no. 23, pp. 3290–3301, 2014.
- [97] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 4. Springer, 2006.
- [98] M. Tsukada, M. Kondo, and H. Matsutani, “A neural network-based on-device learning anomaly detector for edge devices,” *IEEE Transactions on Computers*, vol. 69, no. 7, pp. 1027–1044, 2020.
- [99] G. Wei, J. Yu, J. Wang, P. Gu, D. J. Birch, and Y. Chen, “Hairpin dna-functionalized gold nanorods for mrna detection in homogenous solution,” *Journal of Biomedical Optics*, vol. 21, no. 9, pp. 097001–097001, 2016.
- [100] K. A. Kang, J. Wang, J. B. Jasinski, and S. Achilefu, “Fluorescence manipulation by gold nanoparticles: from complete quenching to extensive enhancement,” *Journal of nanobiotechnology*, vol. 9, no. 1, pp. 1–13, 2011.
- [101] C. Racknor, M. R. Singh, Y. Zhang, D. J. Birch, and Y. Chen, “Energy transfer between a biological labelling dye and gold nanorods,” *Methods and Applications in Fluorescence*, vol. 2, no. 1, p. 015002, 2013.

REFERENCES

- [102] A. Heuer-Jungemann, P. K. Harimech, T. Brown, and A. G. Kanaras, “Gold nanoparticles and fluorescently-labelled dna as a platform for biological sensing,” *Nanoscale*, vol. 5, no. 20, pp. 9503–9510, 2013.
- [103] Y. Zhang, G. Wei, J. Yu, D. J. Birch, and Y. Chen, “Surface plasmon enhanced energy transfer between gold nanorods and fluorophores: application to endocytosis study and rna detection,” *Faraday discussions*, vol. 178, pp. 383–394, 2015.
- [104] Y. Zhang, J. Yu, D. J. Birch, and Y. Chen, “Gold nanorods for fluorescence lifetime imaging in biology,” *Journal of biomedical optics*, vol. 15, no. 2, pp. 020504–020504, 2010.
- [105] T. Durduran and A. G. Yodh, “Diffuse correlation spectroscopy for non-invasive, micro-vascular cerebral blood flow measurement,” *Neuroimage*, vol. 85, pp. 51–63, 2014.
- [106] S. A. Carp, M. B. Robinson, and M. A. Franceschini, “Diffuse correlation spectroscopy: current status and future outlook,” *Neurophotonics*, vol. 10, no. 1, pp. 013509–013509, 2023.
- [107] Y. Shang, T. Li, and G. Yu, “Clinical applications of near-infrared diffuse correlation spectroscopy and tomography for tissue blood flow monitoring and imaging,” *Physiological measurement*, vol. 38, no. 4, p. R1, 2017.
- [108] Y. Shang, T. Symons, T. Durduran, A. G. Yodh, and G. Yu, “Effects of muscle fiber motion on diffuse correlation spectroscopy blood flow measurements during exercise,” *Biomedical optics express*, vol. 1, no. 2, pp. 500–511, 2010.

REFERENCES

- [109] C.-G. Bangalore-Yogananda, R. Rosenberry, S. Soni, H. Liu, M. D. Nelson, and F. Tian, “Concurrent measurement of skeletal muscle blood flow during exercise with diffuse correlation spectroscopy and doppler ultrasound,” *Biomedical Optics Express*, vol. 9, no. 1, pp. 131–141, 2018.
- [110] M. A. Wayne, E. J. Sie, A. C. Ulku, P. Mos, A. Ardelean, F. Marsili, C. Bruschini, and E. Charbon, “Massively parallel, real-time multispeckle diffuse correlation spectroscopy using a 500× 500 spad camera,” *Biomedical Optics Express*, vol. 14, no. 2, pp. 703–713, 2023.
- [111] E. J. Sie, H. Chen, E.-F. Saung, R. Catoen, T. Tiecke, M. A. Chevillet, and F. Marsili, “High-sensitivity multispeckle diffuse correlation spectroscopy,” *Neurophotonics*, vol. 7, no. 3, pp. 035010–035010, 2020.
- [112] C. Zhou, G. Yu, D. Furuya, J. H. Greenberg, A. G. Yodh, and T. Durduran, “Diffuse optical correlation tomography of cerebral blood flow during cortical spreading depression in rat brain,” *Optics express*, vol. 14, no. 3, pp. 1125–1144, 2006.
- [113] C. J. Stapels, N. J. Kolodziejewski, D. McAdams, M. J. Podolsky, D. E. Fernandez, D. Farkas, and J. F. Christian, “A scalable correlator for multichannel diffuse correlation spectroscopy,” in *Advanced Biomedical and Clinical Diagnostic and Surgical Guidance Systems XIV*, vol. 9698, pp. 106–112, SPIE, 2016.
- [114] S. O. Rice, “Mathematical analysis of random noise,” *The Bell System Technical Journal*, vol. 23, no. 3, pp. 282–332, 1944.

REFERENCES

- [115] H. Zhao, E. Sathialingam, and E. M. Buckley, “Accuracy of diffuse correlation spectroscopy measurements of cerebral blood flow when using a three-layer analytical model,” *Biomedical Optics Express*, vol. 12, no. 11, pp. 7149–7161, 2021.
- [116] D. Mazumder, M. M. Wu, N. Ozana, D. Tamborini, M. A. Franceschini, and S. A. Carp, “Optimization of time domain diffuse correlation spectroscopy parameters for measuring brain blood flow,” *Neurophotonics*, vol. 8, no. 3, pp. 035005–035005, 2021.
- [117] W. Lin, D. R. Busch, C. C. Goh, J. Barsi, and T. F. Floyd, “Diffuse correlation spectroscopy analysis implemented on a field programmable gate array,” *IEEE Access*, vol. 7, pp. 122503–122512, 2019.
- [118] P. Zhang, Z. Gui, L. Hao, X. Zhang, C. Liu, and Y. Shang, “Signal processing for diffuse correlation spectroscopy with recurrent neural network of deep learning,” in *2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 328–332, IEEE, 2019.
- [119] C.-S. Poon, F. Long, and U. Sunar, “Deep learning model for ultrafast quantification of blood flow in diffuse correlation spectroscopy,” *Biomedical Optics Express*, vol. 11, no. 10, pp. 5557–5564, 2020.
- [120] Z. Li, Q. Ge, J. Feng, K. Jia, and J. Zhao, “Quantification of blood flow index in diffuse correlation spectroscopy using long short-term memory architecture,” *Biomedical Optics Express*, vol. 12, no. 7, pp. 4131–4146, 2021.

REFERENCES

- [121] J. Feng, M. Jiang, J. Bai, K. Jia, and Z. Li, “Cerebral blood flow monitoring using a convgru model based on diffuse correlation spectroscopy,” *Infrared Physics & Technology*, vol. 129, p. 104541, 2023.
- [122] A. Fog, “Instruction tables-lists of instruction latencies, throughputs and micro-operation breakdowns for intel, amd, and via cpus,” 2022.
- [123] J. Dong, R. Bi, J. H. Ho, P. S. Thong, K.-C. Soo, and K. Lee, “Diffuse correlation spectroscopy with a fast fourier transform-based software autocorrelator,” *Journal of biomedical optics*, vol. 17, no. 9, pp. 097004–097004, 2012.
- [124] S. Xu, X. Yang, W. Liu, J. Jönsson, R. Qian, P. C. Konda, K. C. Zhou, L. Kreiß, H. Wang, Q. Dai, *et al.*, “Imaging dynamics beneath turbid media via parallelized single-photon detection,” *Advanced Science*, vol. 9, no. 24, p. 2201885, 2022.
- [125] J. Buchholz, J. W. Krieger, G. Mocsár, B. Kreith, E. Charbon, G. Vámosi, U. Kebschull, and J. Langowski, “Fpga implementation of a 32x32 autocorrelator array for analysis of fast image series,” *Optics express*, vol. 20, no. 16, pp. 17767–17782, 2012.
- [126] Z. Zang, D. Xiao, Q. Wang, Z. Jiao, Y. Chen, and D. D. U. Li, “Compact and robust deep learning architecture for fluorescence lifetime imaging and fpga implementation,” *Methods and Applications in Fluorescence*, vol. 11, no. 2, p. 025002, 2023.
- [127] D. A. Boas, L. Campbell, and A. G. Yodh, “Scattering and imaging with diffusing temporal field correlations,” *Physical review letters*, vol. 75, no. 9, p. 1855, 1995.

REFERENCES

- [128] M. Seong, Y. Oh, K. Lee, and J. G. Kim, “Blood flow estimation via numerical integration of temporal autocorrelation function in diffuse correlation spectroscopy,” *Computer Methods and Programs in Biomedicine*, vol. 222, p. 106933, 2022.
- [129] R. Alfano, W. Wang, L. Wang, and S. Gayen, “Light propagation in highly scattering turbid media: concepts, techniques, and biomedical applications,” *Photonics: Biomedical Photonics, Spectroscopy, and Microscopy*, pp. 367–412, 2015.
- [130] C. Cheung, J. P. Culver, K. Takahashi, J. H. Greenberg, and A. Yodh, “In vivo cerebrovascular measurement combining diffuse near-infrared absorption and correlation spectroscopies,” *Physics in Medicine & Biology*, vol. 46, no. 8, p. 2053, 2001.
- [131] T. J. Farrell, M. S. Patterson, and B. Wilson, “A diffusion theory model of spatially resolved, steady-state diffuse reflectance for the noninvasive determination of tissue optical properties in vivo,” *Medical physics*, vol. 19, no. 4, pp. 879–888, 1992.
- [132] L. He, Y. Lin, Y. Shang, B. J. Shelton, and G. Yu, “Using optical fibers with different modes to improve the signal-to-noise ratio of diffuse correlation spectroscopy flow-oximeter measurements,” *Journal of biomedical optics*, vol. 18, no. 3, pp. 037001–037001, 2013.
- [133] R. Bi, J. Dong, and K. Lee, “Deep tissue flowmetry based on diffuse speckle contrast analysis,” *Optics letters*, vol. 38, no. 9, pp. 1401–1403, 2013.

REFERENCES

- [134] X. Cheng, H. Chen, E. J. Sie, F. Marsili, and D. A. Boas, “Development of a monte carlo-wave model to simulate time domain diffuse correlation spectroscopy measurements from first principles,” *Journal of Biomedical Optics*, vol. 27, no. 8, pp. 083009–083009, 2022.
- [135] S. A. Carp, D. Tamborini, D. Mazumder, K.-C. Wu, M. R. Robinson, K. A. Stephens, O. Shatrovoy, N. Lue, N. Ozana, M. H. Blackwell, *et al.*, “Diffuse correlation spectroscopy measurements of blood flow using 1064 nm light,” *Journal of Biomedical Optics*, vol. 25, no. 9, pp. 097003–097003, 2020.
- [136] J. D. Johansson, D. Portaluppi, M. Buttafava, and F. Villa, “A multipixel diffuse correlation spectroscopy system based on a single photon avalanche diode array,” *Journal of biophotonics*, vol. 12, no. 11, p. e201900091, 2019.
- [137] Q. Fang and D. A. Boas, “Monte carlo simulation of photon migration in 3d turbid media accelerated by graphics processing units,” *Optics express*, vol. 17, no. 22, pp. 20178–20190, 2009.
- [138] Z. Zang, D. Xiao, Q. Wang, Z. Jiao, Z. Li, Y. Chen, and D. D.-U. Li, “Hardware inspired neural network for efficient time-resolved biomedical imaging,” in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1883–1886, IEEE, 2022.
- [139] “Nvml api reference guide,”
<https://docs.nvidia.com/deploy/nvml-api/index.html>.
- [140] “Xilinx power estimator user guide,” <https://docs.xilinx.com/r/en-US/ug440-xilinx-power-estimator?tocId=nnrf2odl4xIaqGp3~WtIBA>.

REFERENCES

- [141] “pyjoules,” <https://docs.nvidia.com/deploy/nvml-api/index.html>.
- [142] G. Yao, Y. Chen, C. Jiang, Y. Xuan, X. Hu, Y. Liu, and Y. Pan, “Dynamic single-photon 3d imaging with a sparsity-based neural network,” *Optics Express*, vol. 30, no. 21, pp. 37323–37340, 2022.
- [143] Y. Lin and E. Charbon, “Spiking neural networks for active time-resolved spad imaging,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 8147–8156, 2024.