This article has been accepted for publication in IEEE Signal Processing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LSP.2025.3539908

Deep Unrolled Graph Laplacian Regularization for Robust Time-of-Flight Depth Denoising

Jingwei Jia, Changyong He, Jianhui Wang, Gene Cheung, *Fellow, IEEE,* Jin Zeng, *Senior Member, IEEE*

***Abstract*—Depth images captured by Time-of-Flight (ToF) sensors are subject to severe noise. Recent approaches based on deep neural networks achieve good depth denoising performance in synthetic data, but the application to real-world data is limited, due to the complexity of actual depth noise characteristics and the difficulty in acquiring ground truth. In this paper, we propose a novel ToF depth denoising network based on unrolled graph Laplacian regularization to "robustify" the network against both noise complexity and dataset deficiency. Unlike previous schemes that are ignorant of underlying ToF imaging mechanism, we formulate a fidelity term in the optimization problem to adapt to the depth probabilistic distribution with spatially-varying noise variance. Then, we add quadratic graph Laplacian regularization as the smoothness prior, leading to a maximum a posteriori problem that is optimized efficiently by solving a linear system of equations. We unroll the solution into iterative filters so that parameters used in the optimization and graph construction are amendable to data-driven tuning. Because the resulting network is built using domain knowledge of ToF imaging principle and graph prior, it is robust against overfitting to synthetic training data. Experimental results demonstrate that the proposal outperforms existing schemes in ToF depth denoising on synthetic FLAT dataset and generalizes well to real Kinectv2 dataset.**

***Index Terms*—Depth denoising, Time-of-Flight sensor, graph signal processing, deep neural network.**

I. Introduction

DUE to the low-cost CMOS sensor technology and low power requirement, continuous-wave Time-of-Flight (ToF) depth sensors [1] have emerged as an exciting 3D imaging modality empowering various vision applications [2]–[4]. Depth images captured by commercial ToF sensors such as Microsoft Kinect suffer from severe noise on dark, distant, and glossy surfaces [5], motivating various ToF depth denoising schemes [6]–[11]. Model-based methods are based on mathematical models, such as bilateral filter [12] and non-local means [13], [14]. Leveraging progress in *graph signal processing* (GSP) [15], [16], recent schemes construct graphs to encode pixel correlations in depth images, then formulate ToF depth denoising as a *maximum a posteriori* (MAP) problem [17] using graph-based priors. For example, [18] used a sparsity prior in graph transform domain, and [19], [20] adopted smoothness prior via graph Laplacian regularization (GLR) [21]. However, *existing model-based schemes are ignorant of distinctive ToF depth noise statistics, leading to inaccurate optimization formulations and sub-par performance.*

Recent approaches based on *deep neural networks* (DNNs) achieved state-of-the-art ToF denoising performance using synthetic data for training. While many approaches denoise generated depth images [22], [23], errors accumulate during image formation from raw ToF data, resulting in unique depth noise characteristics that hamper denoising performance [7]. More recent methods take raw ToF data as input and build end-to-end networks to produce denoised depth images [7]–[11]. For example, ToFNet [7] generated restored depth from raw data with a multi-scale network, significantly improving imaging quality. However, *existing DNN-based schemes are purely data-driven and do not account for ToF noise patterns, resulting in poor generalization to real data, due to difficulty in acquiring ground truth.* Although domain adaptation has been adopted to enhance network generalization ability [9], the performance still drops significantly at high noise levels.

To address the above issues, we propose the *Graph Laplacian Regularization Unrolling Network* (GLRUN) for robust and interpretable ToF depth denoising. Unlike existing schemes ignorant of the ToF imaging mechanism, we formulate a MAP problem based on an accurate ToF noise model, which is optimally denoised before converting to depth. Different from existing graph-based methods [18]–[20] constructing graphs with hand-crafted features, we unroll GLR-based solution into iterative low-pass graph filters via its diffusion interpretation, so that parameters used in graph construction and optimization are end-to-end trained. Resulting network is built from domain knowledge of ToF imaging principle and GLR prior, which restricts its solution space [24] and makes it more robust against overfitting than existing DNN-based schemes [9]–[11]. Our contributions are as follows.

- We formulate a new MAP problem based on ToF depth noise analysis that denoises raw ToF data with adaptation to spatially-varying depth noise variance;
- We solve the problem with GLR prior by unrolling the solution into iterative graph filters to enable data-driven parameter optimization;
- We interpret GLRUN as a sequence of low-pass graph filtering, which explains its robustness to overfitting.

We demonstrate the enhanced accuracy of GLRUN on the FLAT dataset [8], decreasing MAE by 37.4% over competing schemes. In addition, we show strong generalization ability of

The work was supported in part by National Natural Science Foundation of China under Grant 62201389 and in part by Shanghai Rising-Star Program under Grant 22YF1451200. (*Corresponding author: Jin Zeng.*)

Jingwei Jia, Changyong He, Jin Zeng are with the School of Computer Science and Technology, Tongji University, Shanghai, 201804, China. (e-mail: {1952040, 1952774, zengjin}@tongji.edu.cn)

Jianhui Wang is with the College of Electronic and Information Engineering, Tongji University, Shanghai, 201804, China. (e-mail: 1952934@tongji.edu.cn)

Gene Cheung is with the Department of EECS, York University, Toronto, ON M3J 1P3, Canada. (e-mail: genec@yorku.ca)

Jingwei Jia, Changyong He, and Jianhui Wang contributed equally.

This article has been accepted for publication in IEEE Signal Processing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LSP.2025.3539908

GLRUN to real noise in ToF data captured by Kinectv2 [5].

II. Noise Analysis and Problem Formulation

In this section, we review the ToF imaging mechanism and depth noise statistics, then derive a MAP formulation with a GLR prior to denoise raw ToF data before converting to depth.

A. ToF Imaging Mechanism

To measure depth x_d of an object, the ToF camera emits a periodic signal $s_e(t)$ modulated by a sinusoidal function with frequency f_m and receives the reflected signal $s_r(t)$ with phase shift ϕ after the signal has traveled the distance $2x_d$ [25]. By measuring the correlation between $s_r(t)$ and the phase shifted version of $s_e(t)$ with phase offset θ, the raw measurements are given as $c_\theta = \alpha \cos(\phi + \theta) + \beta$, where α is the signal amplitude, and β is the ambient light intensity. By measuring c_θ for multiple phase offsets θ, the raw ToF pair, *i.e.*, in-phase x_i and quadrature x_q components of ϕ, are computed as [11],

$$x_i = \sum_\theta \cos(\theta) c_\theta,\ x_q = \sum_\theta -\sin(\theta) c_\theta. \tag{1}$$

Then, depth x_d and amplitude x_a are reconstructed as

$$x_d = \frac{c\phi}{4\pi f_m},\ \phi = \arctan(x_q/x_i),\ x_a = \sqrt{x_i^2 + x_q^2}, \tag{2}$$

where c is the light speed.

B. Depth Noise Probability Distribution Model

We study how noises in raw data x_i and x_q affect the depth estimation. It is commonly assumed that the noisy versions of x_i and x_q, *i.e.*, y_i, y_q, are independent and identically distributed with bivariate Gaussian distribution [14], [26],

$$P(y_i, y_q | x_i, x_q) = \frac{1}{2\pi\sigma^2} \exp(-\frac{(x_i - y_i)^2 + (x_q - y_q)^2}{2\sigma^2}), \tag{3}$$

where σ is the noise variance. Under normal noise level, *i.e.*, $\gamma = \sigma / y_a \ll 1$, where y_a is the noisy amplitude, the distribution of depth noise n_d is derived in [14] as

$$P(n_d) \approx \frac{\cos(4\pi f_m n_d / c)}{\gamma\sqrt{2\pi}} \exp(-\frac{\sin^2(4\pi f_m n_d / c)}{2\gamma^2}). \tag{4}$$

(4) is close to a Gaussian distribution with variance approximately proportional to γ / f_m. This requires a denoising scheme to be adaptive to frequency f_m and spatially varying y_a, so we *flexibly adjusts denoising operations according to the spatially-varying noise variance*. Next, we formulate a MAP problem with data fidelity term $P(y_d|x_d)$ and prior term $P(x_d)$ described as follows.

C. Data Fidelity Term

Since raw data y_i, y_q exhibit a simpler noise model than y_d, we optimize x_i, x_q then convert to x_d instead of denoising x_d directly. Based on (4), the log of likelihood $P(y_d|x_d)$ is

$$\ln P(y_d|x_d) \approx \ln(\cos(4\pi f_m n_d/c)) - \sin^2(4\pi f_m n_d/c)/(2\gamma^2) \tag{5}$$

where the constant term $-\ln(\gamma\sqrt{2\pi})$ is removed. Both terms in (5) minimize n_d, and with $\gamma \ll 1$, the second term dominates. Thus, we remove the first term and compute the likelihood as a function of x_i, x_q as follows:

$$\ln P(y_d|x_d) \approx -\sin^2(\phi - \phi')/(2\gamma^2) \tag{6}$$

$$= -(\sin\phi\cos\phi' - \cos\phi\sin\phi')^2/(2\gamma^2), \tag{7}$$

where $\phi' = 4\pi f_m y_d / c$ is the noisy phase. From (2) we have

$$\ln P(y_d|x_d) \approx -(x_q y_i - x_i y_q)^2 / (2\sigma^2\, x_a^2). \tag{8}$$

D. GLR Prior for Raw Data

Since we perform denoising on x_i, x_q, we replace prior $P(x_d)$ with $P(x_i, x_q)$. Denote by $\mathbf{x}_i, \mathbf{x}_q, \mathbf{y}_i, \mathbf{y}_q \in \mathbb{R}^N$ the clean and noisy raw ToF image pairs in vectorized form, respectively, where N is the number of pixels in the image. GLR smoothness prior is widely used for signal recovery [20], [21], [27]–[29], and we adopt GLR prior for $\mathbf{x}_i, \mathbf{x}_q$ so that the optimization is efficiently computed by solving a linear system. Specifically, we model $\mathbf{x}_i, \mathbf{x}_q$ as 8-connected graphs [18] with each pixel connected to its 8 neighbors. Similarities between connected pixel pairs are modeled using graph Laplacian matrices $\mathbf{L}_i, \mathbf{L}_q$ that are symmetric and positive semi-definite (PSD) with positive edge weights [15]. The graph edge weights are end-to-end trained with details discussed in Sec. III-B. GLR prior is given as

$$P(\mathbf{x}_i, \mathbf{x}_q) = \exp(-\frac{\mathbf{x}_i^\top \mathbf{L}_i \mathbf{x}_i + \mathbf{x}_q^\top \mathbf{L}_q \mathbf{x}_q}{\sigma_L^2}), \tag{9}$$

where σ_L adjusts the sensitivity to variations on graphs.

Based on (8) and (9), the MAP problem is optimized as

$$\min_{\mathbf{x}_i, \mathbf{x}_q} \left\| \frac{\mathbf{X}_a^{-1}(\mathbf{x}_q \odot \mathbf{y}_i - \mathbf{x}_i \odot \mathbf{y}_q)}{\sqrt{2}\sigma} \right\|_2^2 + \frac{\mathbf{x}_i^\top \mathbf{L}_i \mathbf{x}_i}{\sigma_L^2} + \frac{\mathbf{x}_q^\top \mathbf{L}_q \mathbf{x}_q}{\sigma_L^2}, \tag{10}$$

where $\mathbf{X}_a = \mathrm{diag}(\mathbf{x}_a)$, $\odot$ is Hadamard product. To solve (10) approximately, we take an alternating approach, where in each iteration, we fix $\mathbf{x}_q$ and solve $\mathbf{x}_i$, then fix $\mathbf{x}_i$ and solve $\mathbf{x}_q$, and repeat until convergence. For example, in iteration l, we compute $\mathbf{x}_a^{l-1}$ based on (2) and set $\mathbf{x}_a = \mathbf{x}_a^{l-1}$, then set $\mathbf{y}_i = \mathbf{x}_i^{l-1}$, $\mathbf{x}_q = \mathbf{y}_q = \mathbf{x}_q^{l-1}$ in (10), and optimize $\mathbf{x}_i^l$ as

$$\min_{\mathbf{x}_i} ||(\mathbf{X}_a^{l-1})^{-1} \mathbf{x}_q^{l-1} \odot (\mathbf{x}_i - \mathbf{x}_i^{l-1})||_2^2 + 2\lambda \mathbf{x}_i^\top \mathbf{L}_i \mathbf{x}_i, \tag{11}$$

where $\lambda = (\sigma/\sigma_L)^2$. Then we set $\mathbf{x}_i = \mathbf{y}_i = \mathbf{x}_i^l$, $\mathbf{y}_q = \mathbf{x}_q^{l-1}$ in (10) and optimize $\mathbf{x}_q^l$. For the first iteration, we set $\mathbf{x}_q^0 = \mathbf{y}_q$, $\mathbf{x}_i^0 = \mathbf{y}_i$. In this way, *we jointly denoise the raw ToF pair* $\mathbf{x}_i, \mathbf{x}_q$ *to optimize depth estimation based on its noise model.* Next, we design an algorithm implementation to solve (10).

III. Algorithm Unrolling and Network Design

In this section, we unroll the solution of (11) into iterative filtering, which is used as the key module to design the proposed GLRUN with graph filtering interpretation.

A. Unrolled GLR Module

The solution of (11) is obtained by solving a linear system:

$$((\mathbf{X}_a^{l-1})^{-1} |\mathbf{x}_q^{l-1}|)^2 \odot (\mathbf{x}_i - \mathbf{x}_i^{l-1}) + 2\lambda \mathbf{L}_i \mathbf{x}_i = 0, \tag{12}$$

where $|\mathbf{x}_q^{l-1}|$ computes element-wise absolute value of $\mathbf{x}_q^{l-1}$. Denote by $\mathbf{W}_i$ and $\mathbf{D}_i = \mathrm{diag}(\mathbf{W}_i \mathbf{1})$ the adjacency and degree

This article has been accepted for publication in IEEE Signal Processing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LSP.2025.3539908

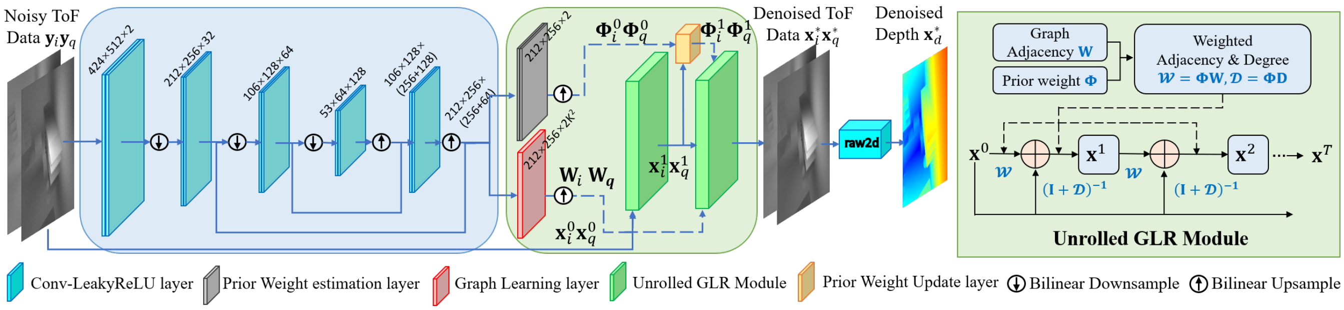

Fig. 1. GLRUN consists of the feature extraction network to estimate initial prior weights and graph adjacency matrices, and Unrolled GLR modules to denoise ToF data. Input dimensions are shown on top of each layer.

matrices of $\mathbf{L}_i$ [15], where $\mathbf{1}$ is an all-ones vector. (12) is rewritten as

$$(\mathbf{x}_i - \mathbf{x}_i^{l-1}) + 2\lambda((|\mathbf{X}_q^{l-1}|)^{-1}\mathbf{X}_a^{l-1})^2(\mathbf{D}_i - \mathbf{W}_i)\mathbf{x}_i = 0, \quad (13)$$

where $\mathbf{X}_q^{l-1} = \text{diag}(\mathbf{x}_q^{l-1})$. $2\lambda((|\mathbf{X}_q^{l-1}|)^{-1}\mathbf{X}_a^{l-1})^2$ can be regarded as the spatially-varying *prior weight*, which we denote as $\mathbf{\Phi}_i^{l-1}$. For accurate estimation of $\mathbf{\Phi}_i^{l-1}$ and $\mathbf{W}_i$, we adopt algorithm unrolling [24] to enable data-driven parameter optimization. While existing algorithm unrolling schemes [30], [31] employ matrix inversion that is not fully trained, we avoid matrix inversion and unroll the solution into iterative filtering so that the parameters are fully trainable.

Specifically, as shown in [32], (13) can be solved using a diffusion scheme based on gradient descent. We adopt the following anisotropic diffusion [33] that runs forward in time on the input $\mathbf{x}_i^{l-1}$ to steady state with initial state $\mathbf{x}_i^{l,0} = \mathbf{x}_i^{l-1}$,

$$\partial_t \mathbf{x}_i = \frac{(\mathbf{x}_i^{l-1} - \mathbf{x}_i) - \mathbf{\Phi}_i^{l-1}(\mathbf{D}_i - \mathbf{W}_i)\mathbf{x}_i}{\mathbf{I} + \mathbf{\Phi}_i^{l-1}\mathbf{D}_i}. \quad (14)$$

Instead of computing the large Hessian matrix for (13) to find an optimal step size for gradient descent [34], we follow [33] and use diagonal $(\mathbf{I} + \mathbf{\Phi}_i^{l-1}\mathbf{D}_i)^{-1}$ as the diffusion coefficient in (14), which is computed efficiently and decreases with the gradient where elements in $\mathbf{\Phi}_i^{l-1}\mathbf{D}_i$ is large. By discretizing $\partial_t \mathbf{x}_i$ with $\mathbf{x}_i^{l,t+1} - \mathbf{x}_i^{l,t}$, (14) becomes

$$\mathbf{x}_i^{l,t+1} - \mathbf{x}_i^{l,t} = \frac{(\mathbf{x}_i^{l-1} - \mathbf{x}_i^{l,t}) - \mathbf{\Phi}_i^{l-1}(\mathbf{D}_i - \mathbf{W}_i)\mathbf{x}_i^{l,t}}{\mathbf{I} + \mathbf{\Phi}_i^{l-1}\mathbf{D}_i}, \quad (15)$$

$$\mathbf{x}_i^{l,t+1} = (\mathbf{I} + \mathbf{\Phi}_i^{l-1}\mathbf{D}_i)^{-1}(\mathbf{x}_i^{l-1} + \mathbf{\Phi}_i^{l-1}\mathbf{W}_i\mathbf{x}_i^{l,t}). \quad (16)$$

From (16), we can see at each time step t, $\mathbf{x}_i^{l,t+1}$ is obtained by the convolutional transform of $\mathbf{x}_i^{l,t}$ with 3×3 kernel specified by $\mathbf{\Phi}_i^{l-1}\mathbf{W}_i$, fused with the initial state $\mathbf{x}_i^{l-1}$. In this way, (11) is solved by recurrently repeating T times of convolutions on $\mathbf{x}_i^{l-1}$, which we refer to as *Unrolled GLR* module, illustrated in the right part of Fig. 1. The notations l and i are eliminated in Fig. 1 since the same procedure applies to each iteration l and the optimization of $\mathbf{x}_q$. Next, we utilize DNN to learn the initial prior weight $\mathbf{\Phi}_i^0$, and graph adjacency $\mathbf{W}_i$.

B. GLRUN Architecture and Graph Filter Interpretation

The proposed GLRUN is illustrated in Fig. 1, which comprises two parts. The first part is the feature extraction network that adopts an encoder-decoder structure with skip-connections [35] to estimate the initial prior weights $\mathbf{\Phi}_i^0, \mathbf{\Phi}_q^0$, and edge weights $\mathbf{W}_i, \mathbf{W}_q$ for the 8-connected graphs for $\mathbf{x}_i, \mathbf{x}_q$. We hereinafter eliminate the notations i and q since the same procedure applies to the two components. We apply sigmoid function on $\mathbf{\Phi}^0$ to get positive weights, then scale by 10 to ensure sufficient denoising strength. To get symmetric and PSD $\mathbf{L}$, we use $\mathbf{W} + \mathbf{W}^\top$ as the new adjacency matrix, then apply softmax function for each pixel to learn positive edge weights. To reduce the computational cost, the outputs are of 1/2 input scale and bilinearly upsampled to match input size.

In the second part, we use two Unrolled GLR modules to denoise raw data, which corresponds to two iterations of optimization in Section II-D. To update prior weights, we set $\mathbf{\Phi}_i^1 = ((\mathbf{X}_i^1)^{-1}\mathbf{X}_q^0)^2\mathbf{\Phi}_i^0$, $\mathbf{\Phi}_q^1 = ((\mathbf{X}_i^1)^{-1}\mathbf{X}_i^0)^2\mathbf{\Phi}_q^0$ based on (13). The final output $\mathbf{x}_i^*$ and $\mathbf{x}_q^*$ are converted to depth $\mathbf{x}_d^*$ via the raw2d module based on (2). In the case of multi-frequency inputs, raw data of different f_m are denoised separately with shared network parameters. Depth maps with different f_m are merged via phase unwrapping [7] to generate the final depth.

Inside the Unrolled GLR module, each convolutional layer computes $\mathbf{\Phi}\mathbf{W}\mathbf{x}^t$. Given non-negative $\mathbf{\Phi}\mathbf{W}$, corresponding to all positive graph weights, each layer corresponds to a one-hop low-pass graph filter [36], and the iterations are repeated until solution convergence. Due to the regularization of the depth noise model and graph prior, GLRUN is fully interpretable as a parameter-optimized low-pass graph filter. This restricts its solution space and makes it less prone to overfitting.

C. Loss Function

We train our network with l_1 loss function supervised by the ground truth $\mathbf{x}_i^{\text{gt}}$, $\mathbf{x}_q^{\text{gt}}$ and $\mathbf{x}_d^{\text{gt}}$ as follows:

$$L = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\sum_{\theta\in\{i,q,d\}}|\mathbf{x}_\theta^*(v) - \mathbf{x}_\theta^{\text{gt}}(v)| \quad (17)$$

where v, $\mathcal{V}$ and $|\mathcal{V}|$ denote the pixel index, set of valid pixels in GT, and the number of valid pixels, respectively. Loss on $\mathbf{x}_d$ is also included since the final target is to reconstruct depth.

IV. Experimental Results

We experimentally validate the effectiveness of GLRUN in ToF depth denoising via comparison with competing schemes on FLAT dataset [8] and real Kinectv2 data.

A. Experiment Setting

Dataset FLAT dataset [8] was used for training and testing, which is a synthetic dataset with simulated Kinect noise,

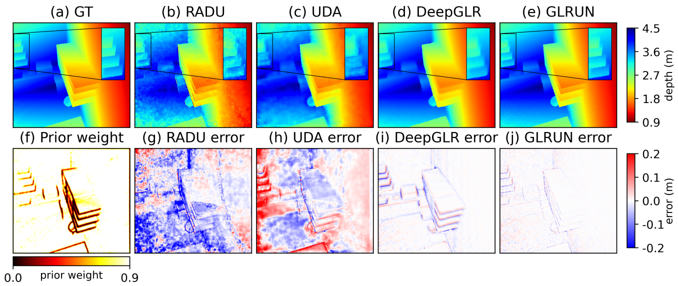

Fig. 2. Depth results and error maps of ToF depth denoised on FLAT dataset [8]: (a) GT, results of (b) RADU [11], (c) UDA [9], (d) DeepGLR [30] and (e) proposed GLRUN. Corresponding error maps are in the second row. (f) shows prior weight for 58MHz ToF data adaptive to input features.

containing 1923 depth images and corresponding raw ToF data. We used 1815 images for training and 108 for testing with image size 424×512. We further collected real Kinectv2 data to test the generalization ability to real ToF data.

Training Details We used Adam optimizer with initial learning rate $1e^{-3}$ and decay at epoch $[10, 20, 30, 40]$ with decay rate 0.5. The model was trained from scratch for 50 epochs. We employed the PyTorch framework [37] on a single GeForce RTX 3090 GPU. We set $T = 3$ in each Unrolled GLR module.

Metrics Following [38], we used Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), inverse RMSE (iRMSE), and inverse MAE (iMAE) to evaluate depth estimation, and used average runtime and GPU memory cost for each image on 3090 GPU for complexity comparison.

B. Comparison with Existing Schemes

We compared with depth-based approach DeepToF [22], and raw-based approaches ToFNet [7], UDA [9], RADU [11], and graph-based DeepGLR [30]. The networks were trained on FLAT dataset with codes released by the authors. We also included model-based libfreenect2 [6] without deep learning. We compared both accuracy and complexity with FLAT testing dataset. In addition, we followed [39] to augment FLAT with simulated edge noise. Note that the same model was used for testing in both noise settings to test generalization ability to unseen noise. In Table I, GLRUN achieved the best accuracy among all the methods in both noise settings, decreasing MAE by at least 37.4% over competing schemes with lower complexity than the state-of-the-art UDA and DeepGLR.

Visual comparison in Fig. 2 further validated the advantage of *graph-based methods* over existing methods in preserving sharp details with noise removal. GLRUN surpassed DeepGLR due to the adaptive prior weight shown in Fig. 2(f) and the end-to-end training of parameters used in the optimization.

C. Ablation Study

To investigate the effectiveness of each component in GLRUN, we tested on FLAT dataset with different variants of GLRUN. Specifically, we removed Unrolling GLR modules (UGLR) and reduced the network to a simple UNet as the baseline model. Then, we solved the MAP problem via either the algorithm in DeepGLR [30] or our UGLR without prior weight layer ($\mathbf{\Phi}$). In addition, we investigated the effect of $\mathbf{\Phi}$ and the loss of depth and IQ, respectively. Results in Table. II show that each component is essential for denoising accuracy. While using IQ loss generated competitive results by generating accurate $\mathbf{x}_i^*$, $\mathbf{x}_q^*$, including depth loss further refined the final depth output $\mathbf{x}_d^*$. Moreover, while using DeepGLR generated competitive results, using UGLR greatly reduced runtime as shown in Table I by avoiding matrix inversion, and enhanced the accuracy by making parameters fully trainable.

TABLE I
COMPARISON OF COMPUTATIONAL COMPLEXITY AND DENOISING ACCURACY ON FLAT TESTING DATASET AND AUGMENTED DATASET

Method	Runtime (s)	Memory (MB)	FLAT		FLAT with edge noise	
			RMSE(m)	MAE(m)	RMSE(m)	MAE(m)
libfreenect2	-	-	0.4154	0.0798	0.4709	0.1145
DeepToF	0.011	1660	0.1120	0.0490	0.1693	0.0783
ToFNet	0.012	3168	0.1744	0.1176	0.1755	0.1180
RADU	193.317	23478	0.1804	0.1197	0.1632	0.0913
UDA	0.011	1920	0.0681	0.0470	0.1404	0.0495
DeepGLR	35.630	1860	0.0954	0.0171	0.1366	0.0177
GLRUN	0.011	1680	**0.0677**	**0.0107**	**0.1296**	**0.0140**

TABLE II
COMPARISON OF QUANTITATIVE EVALUATION ON FLAT TESTING DATASET WITH GLRUN VARIANTS

Modules			RMSE	MAE	iRMSE	iMAE
GLR	$\mathbf{\Phi}$	loss	(m)	(m)	(1/m)	(1/m)
-	-	depth+IQ	0.0809	0.0140	0.0625	0.0162
DeepGLR	-	depth+IQ	0.0784	0.0147	0.0552	0.0124
UGLR	-	depth+IQ	0.0748	0.0145	0.0541	0.0139
UGLR	✓	depth	0.0717	0.0137	0.0547	0.0150
UGLR	✓	IQ	0.0688	0.0112	0.0336	0.0087
UGLR	✓	depth+IQ	**0.0677**	**0.0107**	**0.0304**	**0.0084**

D. Generalization Ability Evaluation with Real Kinectv2 Data

We captured real ToF data with Kinectv2 sensor and conducted qualitative comparison shown in Fig. 3. The same model trained on synthetic FLAT dataset is used for testing on real data. DNN-based methods UDA and RADU generated blurry results. This was due to the poor generalization to real data with different noise characteristics from synthetic training data. GLRUN showed strong generalization ability and better detail preservation due to the specified low-pass graph filter.

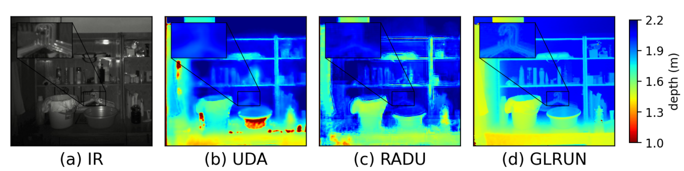

Fig. 3. Visual results of ToF depth denoising on real Kinectv2 data: (a) IR image and results of (b) UDA [9], (c) RADU [11], and (d) GLRUN.

V. CONCLUSION

In this paper, we propose GLRUN for ToF depth denoising that is robust to complicated noise characteristics and training data insufficiency. Based on ToF depth noise model analysis, we propose graph-based MAP problem formulation to optimize raw ToF data. The optimization is implemented via iterative diffusion to incorporate with DNN and enable data-driven parameter optimization. The resulting network shows enhanced denoising accuracy on synthetic data and higher robustness to real noise over competing schemes due to the graph filter interpretation.

This article has been accepted for publication in IEEE Signal Processing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LSP.2025.3539908

References

[1] A. Bhandari and R. Raskar, "Signal processing for time-of-flight imaging sensors: An introduction to inverse problems in computational 3-d imaging," *IEEE Signal Processing Magazine*, vol. 33, no. 5, pp. 45–58, 2016.

[2] J. Kang, S. Lee, M. Jang, and S. Lee, "Gradient flow evolution for 3d fusion from a single depth sensor," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 4, pp. 2211–2225, 2021.

[3] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[4] R. Du, E. Turner, M. Dzitsiuk, L. Prasso, I. Duarte, J. Dourgarian, J. Afonso, J. Pascoal, J. Gladstone, N. Cruces *et al.*, "Depthlab: Real-time 3d interaction with depth maps for mobile augmented reality," in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, 2020, pp. 829–843.

[5] G. Kurillo, E. Hemingway, M.-L. Cheng, and L. Cheng, "Evaluating the accuracy of the Azure Kinect and Kinect v2," *Sensors*, vol. 22, no. 7, p. 2469, 2022.

[6] L. Xiang, F. Echtler, C. Kerl, T. Wiedemeyer, R. Gordon, and F. Facioni, "libfreenect2: Release 0.2," 2016.

[7] S. Su, F. Heide, G. Wetzstein, and W. Heidrich, "Deep end-to-end time-of-flight imaging," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6383–6392.

[8] Q. Guo, I. Frosio, O. Gallo, T. Zickler, and J. Kautz, "Tackling 3d tof artifacts through learning and the flat dataset," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 368–383.

[9] G. Agresti, H. Schaefer, P. Sartor, and P. Zanuttigh, "Unsupervised domain adaptation for tof data denoising with adversarial learning," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5579–5586.

[10] Y. Chen, J. Ren, X. Cheng, K. Qian, L. Wang, and J. Gu, "Very power efficient neural time-of-flight," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2257–2266.

[11] M. Schelling, P. Hermosilla, and T. Ropinski, "Radu: Ray-aligned depth update convolutions for tof data denoising," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 671–680.

[12] J. T. Barron and B. Poole, "The fast bilateral solver," in *European conference on computer vision*. Springer, 2016, pp. 617–632.

[13] M. Frank, M. Plaue, and F. A. Hamprecht, "Denoising of continuous-wave time-of-flight depth images using confidence measures," *Optical Engineering*, vol. 48, no. 7, pp. 077 003–077 003, 2009.

[14] M. Georgiev, R. Bregović, and A. Gotchev, "Time-of-flight range measurement in low-sensing environment: Noise analysis and complex-domain non-local denoising," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 2911–2926, 2018.

[15] G. Cheung, E. Magli, Y. Tanaka, and M. K. Ng, "Graph spectral image processing," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 907–930, 2018.

[16] A. Ortega, P. Frossard, J. Kovavcević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[17] G. Sagi and T. Routtenberg, "MAP estimation of graph signals," *IEEE Transactions on Signal Processing*, vol. 72, pp. 463–479, 2024.

[18] W. Hu, X. Li, G. Cheung, and O. Au, "Depth map denoising using graph-based transform and group sparsity," in *2013 IEEE 15th international workshop on multimedia signal processing (MMSP)*. IEEE, 2013, pp. 001–006.

[19] M. Rossi, M. E. Gheche, A. Kuhn, and P. Frossard, "Joint graph-based depth refinement and normal estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 154–12 163.

[20] X. Zhang, G. Cheung, J. Pang, Y. Sanghvi, A. Gnanasambandam, and S. H. Chan, "Graph-based depth denoising & dequantization for point cloud enhancement," *IEEE Transactions on Image Processing*, vol. 31, pp. 6863–6878, 2022.

[21] J. Pang and G. Cheung, "Graph laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770–1785, 2017.

[22] J. Marco, Q. Hernandez, A. Munoz, Y. Dong, A. Jarabo, M. H. Kim, X. Tong, and D. Gutierrez, "Deeptof: off-the-shelf real-time correction of multipath interference in time-of-flight imaging," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 6, pp. 1–12, 2017.

[23] S. Yan, C. Wu, L. Wang, F. Xu, L. An, K. Guo, and Y. Liu, "Ddrnet: Depth map denoising and refinement for consumer depth cameras using cascaded cnns," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 151–167.

[24] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[25] P. Zanuttigh, G. Marin, C. Dal Mutto, F. Dominio, L. Minto, G. M. Cortelazzo *et al.*, "Time-of-flight and structured light depth cameras," *Technology and Applications*, pp. 978–3, 2016.

[26] M. Frank, M. Plaue, H. Rapp, U. Köthe, B. Jähne, and F. A. Hamprecht, "Theoretical and experimental error analysis of continuous-wave time-of-flight range cameras," *Optical Engineering*, vol. 48, no. 1, pp. 013 602–013 602, 2009.

[27] A. Elmoataz, O. Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1047–1060, 2008.

[28] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1327–1336, 2011.

[29] J. Wang, L. Sun, R. Xiong, Y. Shi, Q. Zhu, and B. Yin, "Depth map super-resolution based on dual normal-depth regularization and graph laplacian prior," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 3304–3318, 2022.

[30] J. Zeng, J. Pang, W. Sun, and G. Cheung, "Deep graph laplacian regularization for robust denoising of real images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[31] R. Yoshida, K. Kodama, H. Vu, G. Cheung, and T. Hamamoto, "Unrolling graph total variation for light field image denoising," in *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2022, pp. 2162–2166.

[32] D. M. Strong and T. F. Chan, "Spatially and scale adaptive total variation based regularization and anisotropic diffusion in image processing," in *Diusion in Image Processing, UCLA Math Department CAM Report*. Citeseer, 1996.

[33] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

[34] S. Bubeck *et al.*, "Convex optimization: Algorithms and complexity," *Foundations and Trends® in Machine Learning*, vol. 8, no. 3-4, pp. 231–357, 2015.

[35] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[36] A. Ortega, *Introduction to graph signal processing*. Cambridge University Press, 2022.

[37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[38] Y. Wang, B. Li, G. Zhang, Q. Liu, T. Gao, and Y. Dai, "Lrru: Long-short range recurrent updating networks for depth completion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9422–9432.

[39] J. T. Barron and J. Malik, "Intrinsic scene properties from a single rgb-d image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 17–24.