Supplementary Material for "Exploiting Dual-Correlation for Multi-frame Time-of-Flight Denoising"

Guanting Dong, Yueyi Zhang✉, Xiaoyan Sun, and Zhiwei Xiong

MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition,
University of Science and Technology of China
gtdong@mail.ustc.edu.cn, {zhyuey, sunxiaoyan, zwxiong}@ustc.edu.cn

1 Residual Channel Attention Block

In the main manuscript, we introduced the Residual Channel Attention Block (RCAB) within the context of the Residual Combination Operation for depth prediction and the Feature Extractor during the confidence-guided residual pyramid estimation phase. Here, we delve deeper into the details of RCAB and present ablation experiments to analyze its impact.

Architecture of RCAB. We employ the RCAB to extract the features from the concatenation of depth and amplitude maps for balancing the time consumption and ToF denoising performance. In addition, different from SHARP-Net [2], we incorporate a channel attention layer to combine the estimated confidence-guided residual pyramid instead of a 1×1 convolution layer.

The detailed architecture of the RCAB is shown in Fig. 1. For the Feature Extractor, the RCAB takes the features from the above block F as input to obtain the aggregated feature $F^{'}$. In the Residual Combination Operation, the RCAB takes as input the concatenation of the dot product of the residual pyramid and the confidence pyramid at each level. RCAB outputs a coarse depth residual R_{coarse}, which is employed to remove the MPI and shot noise in ToF depth images.

Ablation study. To validate the effectiveness of RCAB, we conduct experiments to compare our method against its variants, as shown in Table 1. We create three variants by removing RCAB from our network: 'FE w/o RCAB', 'RCO w/o RCAB', and 'Ours w/o RCAB'. "FE w/o RCAB" refers to a variant of our network that includes a feature extractor of 3×3 convolutions. "FE w/o RCAB" also describes a variant of our network that employs a 1×1 convolution to combine the confidence-guided residual pyramid. 'Ours w/o RCAB' represents a variant of our network without RCAB. By comparing these variants with our method across all error levels, we observe that introducing RCAB is more effective in regions with low error levels and plays a crucial role in high-error areas with geometric details.

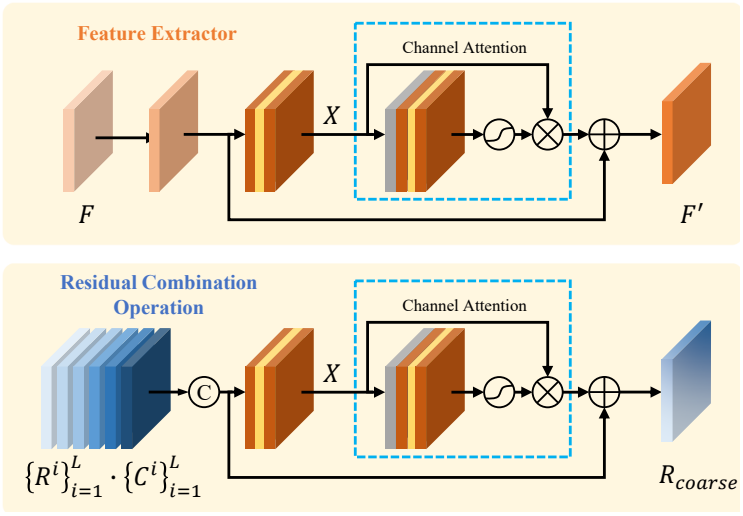

Fig. 1: The architecture of the RCAB in Feature Extractor and Residual Combination Operation. Here, '©' is the concatenation operation. '⊕' and '⊗' represent the addition and multiply operations, respectively. '~' denotes a sigmoid activation function.

Table 1: Quantitative comparison with the variants on the TFT3D dataset.

Model	TFT3D Dataset:MAE(cm)				
	1st Quan.	2nd Quan.	3rd Quan.	4th Quan.	Overall
FE w/o RCAB	0.44	0.51	0.65	2.16	0.94
RCO w/o RCAB	0.42	0.48	0.57	2.21	0.92
Ours w/o RCAB	0.50	0.59	0.69	2.18	0.99
Ours	**0.36**	**0.40**	**0.49**	**2.03**	**0.82**

Table 2: Quantitative comparison with different values of L on the TFT3D dataset.

Model	1st Quan.	2nd Quan.	3rd Quan.	4th Quan.	Overall
Ours ($L = 3$)	0.55	0.59	0.70	2.64	1.12
Ours ($L = 4$)	0.49	0.54	0.66	2.55	1.06
Ours ($L = 5$)	0.46	0.51	0.63	2.44	1.01
Ours ($L = 6$)	0.36	0.40	0.49	2.03	0.82
Ours ($L = 7$)	0.34	0.37	0.48	2.01	0.80
Ours ($L = 8$)	0.34	0.36	0.48	1.98	0.79

2 Depth Refinement Module and Kernel Prediction Network

Following the previous processing steps, MPI noise is significantly reduced. Although shot noise is also mitigated to some extent, it is not as effectively re-

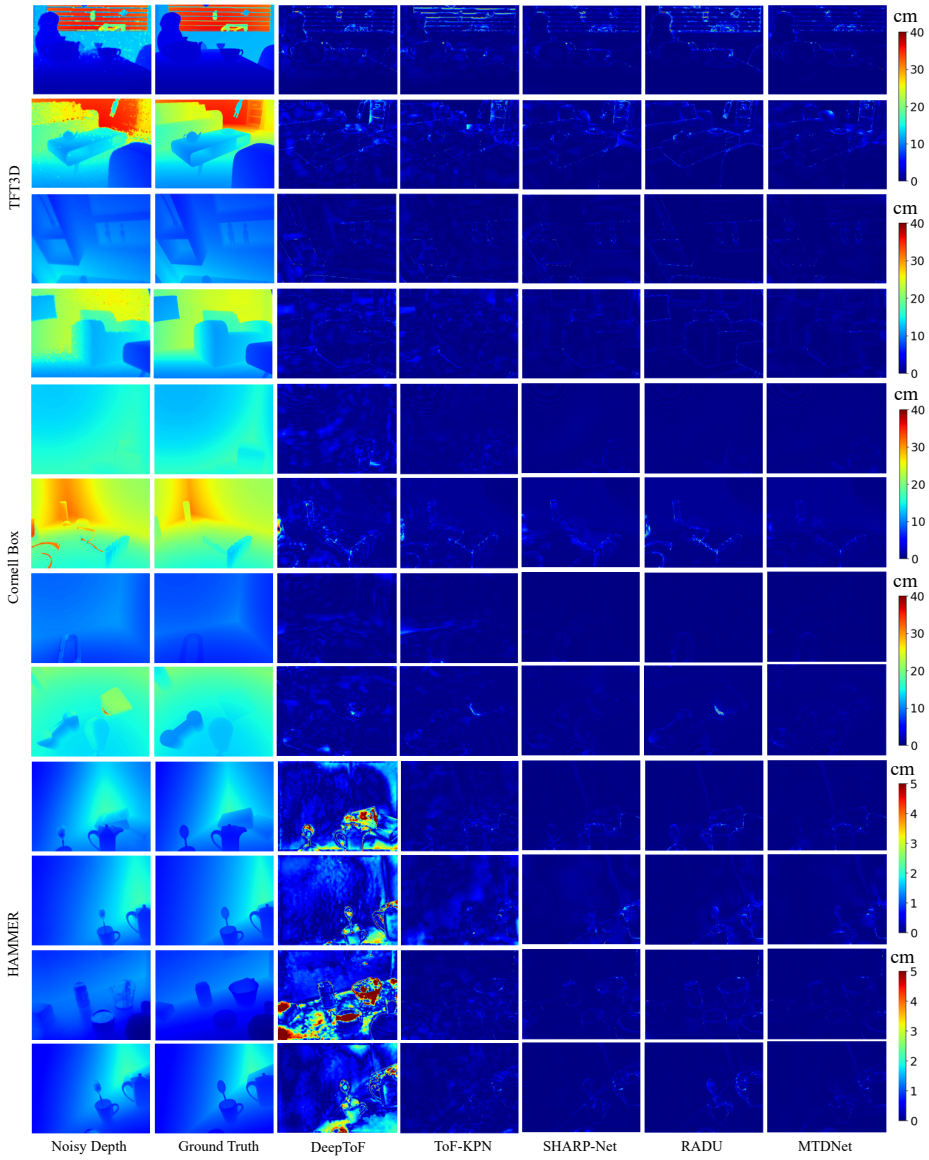

Fig. 2: Qualitative comparison on the TFT3D dataset, the Cornell-Box dataset and the HAMMER iToF dataset for ToF depth denoising. For each dataset, four scenes are selected for comparison. The colour bars on the right show the colour scale for error maps with the unit in cm.

Table 3: The detailed architecture of the DRM.

Module Name	Layer Name	Kernel Size	Stride	Input Channels	Output Channels	Input Layer
Depth Refinement Module (DRM)	conv1_1	3×3	1	1	16	RFM$\bigoplus$depth
	conv1_2	3×3	1	16	16	conv1_1
	conv2_1	3×3	2	16	32	conv1_2
	conv2_2	3×3	1	32	32	conv2_1
	conv3_1	3×3	2	32	64	conv2_2
	conv3_2	3×3	1	64	64	conv3_1
	conv4_1	3×3	2	64	128	conv3_2
	conv4_2	3×3	1	128	128	conv4_1
	upconv1_1	3×3	2	128	64	conv4_2
	upconv1_2	3×3	1	128	64	conv3_2Ⓒupconv1_1
	upconv2_1	3×3	2	64	32	upconv1_2
	upconv2_2	3×3	1	64	32	conv2_2Ⓒupconv2_1
	upconv3_1	3×3	2	32	16	upconv2_2
	upconv3_2	3×3	1	32	16	conv1_2Ⓒupconv3_1
	w	3×3	1	16	9	upconv3_2

moved as MPI. Shot noise still poses a challenge to applying ToF depth sensing. To tackle this issue, we introduce a depth refinement module that leverages a kernel prediction network to output a final denoised ToF depth image denoted as D_{out} [1, 6].

The Depth Refinement Module takes the intermediate depth image as the input and employs a U-Net model with a skip connection to generate a weight matrix. The weight matrix consists of a vectorized filter kernel for each pixel in the depth image. In our experiment, we set the kernel size k as 3, and the size of the weight matrix is $W \times H \times 9$. Next, we generate a patch matrix by vectoring a neighbourhood for each pixel in the depth image. Then the weight matrix is multiplied element-wisely with the patch matrix, generating a 3D volume with the same size. By summing over the 3D volume, we finally get the refined depth image D_{out}. The details of the Depth Refinement Module are shown in Table 3. The $\bigoplus$ symbol and the symbol represent the addition and concatenation operations, respectively.

3 Ablation on the number of pyramid network levels

To determine suitable values for the number of pyramid network levels during training and inference, we compare them with different values of L as shown in Table 2. We observe a consistent decrease in Mean Absolute Error (MAE) across all error levels as the number of pyramid levels increases. This improvement stems from the network's ability to partition the scene into a more fine-grained hierarchical structure with higher pyramid levels. This finer partitioning facilitates a more accurate estimation of MPI noise. To balance runtime efficiency and denoising performance for our proposed method, we ultimately set the number of pyramid levels L to 6.

4 Additional Visualization Results

We present additional error maps in Fig. 2 for a comprehensive comparison of our approach with ToF-KPN [7], SHARP-Net [2], and RADU [8] on both synthetic and realistic datasets. These datasets include the ToF-FlyingThings3D (TFT3D) [7], Cornell-Box [8], and HAMMER [3] datasets. Visual inspection of the error maps reveals that our method performs better, exhibiting smaller errors than the other methods.

Table 4: Quantitative comparison with competitive ToF depth denoising methods on the iToF set and dToF set of the HAMMER dataset.

Model	HAMMER iToF Dataset: MAE (cm)				
	1st Quan.	2nd Quan.	3rd Quan.	4th Quan.	Overall
DeepToF [5]	0.097	0.111	0.138	0.222	0.142
ToF-KPN [7]	0.076	0.087	0.112	0.229	0.126
SHARP-Net [2]	0.048	0.052	0.067	0.149	0.079
RADU [8]	0.049	0.055	0.076	0.204	0.096
Ours	**0.035**	**0.039**	**0.048**	**0.094**	**0.054**
Model	HAMMER dToF Dataset: MAE (cm)				
	1st Quan.	2nd Quan.	3rd Quan.	4th Quan.	Overall
DeepToF [5]	0.094	0.095	0.112	0.175	0.119
ToF-KPN [7]	0.082	0.081	0.092	0.557	0.203
SHARP-Net [2]	0.060	0.060	0.067	0.129	0.079
RADU [8]	0.066	0.066	0.076	0.224	0.108
Ours	**0.056**	**0.057**	**0.065**	**0.106**	**0.071**

5 Quantitative Comparison on direct ToF data

In addition, we evaluate the denoising performance of our method on direct ToF (dToF) data. Although dToF also suffers from MPI, the main source of dToF noise is edge fattening caused by flying pixels, not MPI. We test our method and existing methods on a realistic dToF dataset, *i. e.*, the dToF set of the HAMMER dataset. The experimental results for all the methods on the iToF and dToF sets of the HAMMER dataset are shown in Table 4. We cannot create amplitude maps because the dToF dataset does not have correlation maps. We use the dataset's instance labels as our network's input instead of the amplitude maps. To maintain fairness in comparisons, we adopt the concatenation of an instance label map and a ToF depth map as inputs to our network for the iToF dataset. Compared with RADU and SHARP-Net in all quantiles, our method still achieves optimal performance at all error levels of dToF data. It is observed that while our proposed method outperforms baselines on the dToF set as well, its superiority is less pronounced than it is on iToF data. This difference indicates that the efficacy of our proposed method is intimately linked to the noise composition within the ToF data.

6 Evaluation on temporal continuity

Quantifying temporal continuity metrics is challenging due to our dataset's absence of ground truth for scene flow. To address this limitation, we reference [4] for comparing our method and SHARP-Net using x-t plots, demonstrating improved temporal continuity, as depicted in the figure below.

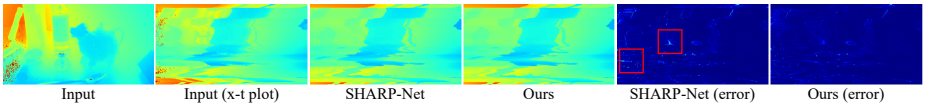

Fig. 3: Visual comparisons with SHARP-Net on the x-t plot of the HAMMER dataset. The red boxes highlight the regions that existing temporal discontinuity.

7 Generalizability on different multi-frame settings

Based on experiments conducted on the HAMMER dataset, depicted below, we observe that increasing the number of frames leads to improved denoising performance of the network. Remarkably, even with just two frames as input, our framework effectively handles the ToF noise.

Table 5: Quantitative comparison with various multi-frame settings on the HAMMER dataset.

# frames	1st Quan.	2nd Quan.	3rd Quan.	4th Quan.	Overall
2	0.050	0.055	0.076	0.207	0.097
3	0.051	0.055	0.074	0.196	0.094
4	0.049	0.054	0.073	0.192	0.093

8 Runtime of different methods

We report the runtime and memory for all methods on a single GTX 1080Ti, as shown in the table below.

Table 6: Quantitative comparison on runtime and memory.

Method	MAE [cm]	Time [ms]	Param. [M]
DeepToF	3.54	243	2.6
ToF-KPN	2.38	359	2.6
SHARP-Net	1.19	404	2.1
RADU	1.28	3650	2.4
Ours	0.82	509	2.9

References

1. Bako, S., Vogels, T., McWilliams, B., Meyer, M., Novák, J., Harvill, A., Sen, P., Derose, T., Rousselle, F.: Kernel-predicting convolutional networks for denoising monte carlo renderings. ACM Transactions on Graphics (TOG) **36**(4), 97 (2017)
2. Dong, G., Zhang, Y., Xiong, Z.: Spatial hierarchy aware residual pyramid network for time-of-flight depth denoising. In: ECCV. pp. 35–50. Springer (2020)
3. Jung, H., Ruhkamp, P., Zhai, G., Brasch, N., Li, Y., Verdie, Y., Song, J., Zhou, Y., Armagan, A., Ilic, S., et al.: Is my depth ground-truth good enough? hammer–highly accurate multi-modal dataset for dense 3d scene regression. arXiv preprint arXiv:2205.04565 (2022)
4. Luo, X., Huang, J.B., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. ACM Transactions on Graphics (TOG) **39**(4), 71–1 (2020)
5. Marco, J., Hernandez, Q., Munoz, A., Dong, Y., Jarabo, A., Kim, M.H., Tong, X., Gutierrez, D.: Deeptof: off-the-shelf real-time correction of multipath interference in time-of-flight imaging. ACM Transactions on Graphics (TOG) **36**(6), 219 (2017)
6. Mildenhall, B., Barron, J.T., Chen, J., Sharlet, D., Ng, R., Carroll, R.: Burst denoising with kernel prediction networks. In: CVPR. pp. 2502–2510 (2018)
7. Qiu, D., Pang, J., Sun, W., Yang, C.: Deep end-to-end alignment and refinement for time-of-flight rgb-d module. In: ICCV. pp. 9994–10003 (2019)
8. Schelling, M., Hermosilla, P., Ropinski, T.: Radu: Ray-aligned depth update convolutions for tof data denoising. In: CVPR. pp. 671–680 (2022)