Supplemental Materials - Efficient Visual Computing with Camera RAW Snapshots

Zhihao Li, Ming Lu, Xu Zhang, Xin Feng, M. Salman Asif, and Zhan Ma

***Abstract*—In this supplementary material, we provide additional information to further evidence the generalization of the proposed ρ-*Vision* for various functionalities. Specifically, we first compare the RGB-Vision and ρ-Vision frameworks using a real-world hardware implementation in Sec. S.I. Then, we provide details of our Unpaired CycleR2R in Sec. S.II and give proofs of some equations in Sec. S.III. In addition, we demonstrate the advantages of running classification and segmentation in the RAW domain directly in Sec. S.IV and Sec. S.V, respectively. At last, we show more visualization results in Sec. S.VI.**

***Index Terms*—Camera RAW, RAW-domain Object Detection, RAW Image Compression**

S.I. A Real-World Hardware Implementation

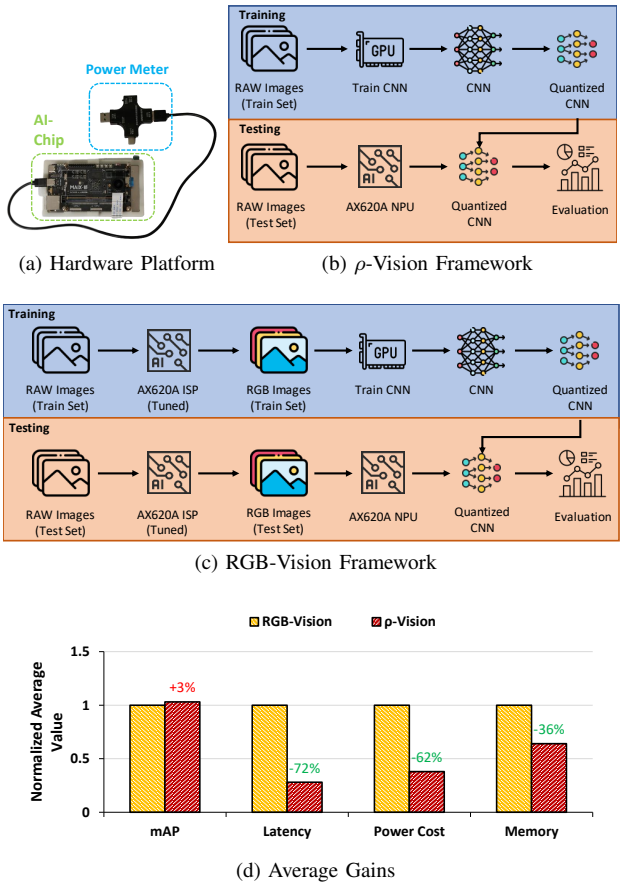
A. Hardware System for Comparative Benchmark

A commodity hardware platform is used to assess the efficiency of RAW-domain visual computing as illustrated in Fig. S1a. It is built upon the Axera-Tech AX620A SoC with a quad-core Arm Cortex-A7 processor, an NPU (Neural Processing Unit), an ISP (Image Signal Processor), and other subsystems. This AX620A SoC is primarily used to process images and videos for vision tasks. Its ISP has two modes: one is the Standard mode (AX620A ISP), and the other is the AI mode (AX620A AI ISP). When using AX620A AI ISP, onboard NPU is utilized to run various neural algorithms like NN (Neural Network) denoising, by which AX620A SoC claims its outstanding performance for low-light imaging.

We use the same RAW samples in the MultiRAW dataset for a fair evaluation. The YOLOv8-S, recommended by the AX620A SoC specification, exemplifies the detection task. Its default settings are assumed for consistency and reproducibility. Upon completing the training of YOLOv8-S, its model is quantized into INT-8 precision using AX620A's official quantization tool and subsequently deployed on AX620A's NPU for inference.

Metrics such as mAP, latency, power consumption, and memory usage are collected for quantitative comparison. With this aim, when executing the YOLOv8-S, a UC96B power meter is connected to the AX620A SoC to collect the power usage, latency is measured using a timer library (C++), and the memory consumption is reported using the default memory monitoring tool provided by the AX620A SoC.

- **ρ-Vision** trains YOLOv8-S using RAW samples (from the iPhone XSmax, a subset of the MultiRAW dataset). Then, such a RAW-domain YOLOv8-S is quantized using the abovementioned rules and deployed on the NPU for detection. For task inference, RAW images are fed directly to the neural model (without requiring ISP computations). Following the common practice, 70 % RAW images are used to train RAW-domain YOLOv8-S, and the remaining 30 % RAW images are tested using quantized YOLOv8-S on NPU. Fig. S1b plots the processing steps in ρ-Vision.
- **RGB-Vision** applies the AX620A ISP onboard to convert RAW images to their corresponding RGB formats for subsequent computations. The training and testing split is the same as in the ρ-Vision paradigm. The RGB-vision



(a) Hardware Platform (b) ρ-Vision Framework

(c) RGB-Vision Framework

(d) Average Gains

Fig. S1: **RGB-Vision vs. ρ-Vision.** (a) The hardware system uses AX620A AI SoC. A UC96B power meter is connected for measurement; (b) ρ-Vision framework trains and tests models using RAW images directly, completely bypassing the ISP; (c) Traditional RGB-Vision framework requires the ISP to generate RGB images for model training and testing; (d) **Average Gains of ρ-Vision to RGB-Vision.** Metrics are normalized to the results generated by the RGB-Vision pipeline.

iPhone → AX620A (D) | iPhone → AX620A (T) | AX620A (T) → AX620A (T) | iPhone → iPhone

Fig. S2: **Impact of ISP used in RGB-Vision on the detection task.** The setup of "Training ISP→Testing ISP" indicates the "Training ISP" used to generate RGB images for training and the "Testing ISP" used to generate RGB images for testing respectively. Default parameters used by the ISP are marked with "(D)" and expert-tuned parameters used by the ISP are annotated with "(T)". The first two columns illustrate domain discrepancies when training and testing using different ISPs, while the last two columns demonstrate how ISP quality (with expert tuning) affects object detection accuracy. *Zoom for better details.*

TABLE S1: Detection performance for various ISP combinations.

Domain	Training ISP	Testing ISP	Car	Person	Traffic Light	Traffic Sign	mAP
RGB-Vision	iPhone	AX620A (Default)	0.324	0.022	0.134	0.213	0.173
	iPhone	AX620A (Tuned)	0.696	0.108	0.523	0.253	0.397
	AX620A (Tuned)	AX620A (Tuned)	0.788	0.225	0.661	0.443	0.529
	iPhone	iPhone	**0.798**	0.219	**0.693**	0.474	**0.546**
ρ-Vision	-	-	0.796	**0.241**	0.655	**0.490**	**0.546**

processing pipeline is pictured in Fig. S1c.

All associated hardware drivers, system images, benchmark code, and datasets will soon be available at https://njuvision.github.io/rho-vision to encourage reproducible research.

B. Experimental Analysis

Overall Evaluation. Fig. S1d showcases the efficacy of the proposed ρ-Vision paradigm. Compared to RGB-Vision, it provides a notable 3 % detection accuracy increase. The same YOLOv8-S is just retrained using RAW images without any dedicated network model engineering. It reduces the latency by 72 %, a critical advancement for autonomous driving applications. Furthermore, the 62 % reduction in power consumption presents significant advantages of ρ-Vision for AIoT devices, where energy efficiency is crucial. The 36 % decrease in memory usage also enables the deployment of ρ-Vision on lower-cost embedded devices. The performance improvement owes to better-preserving scene information in the RAW domain. The skipping of ISP generally avoids the extra computations and memory caching, leading to a noticeable cost and latency reduction. These promise the encouraging potential of ρ-Vision in advancing computer vision applications for better task performance, faster response, and less cost.

Impact of ISP used in RGB-Vision Paradigm. In Fig. S1c, the AX620A ISP is expert-tuned. This is because default settings used in AX620A ISP cannot provide a decent result, which motivates us to study the impact of various ISP configurations on task efficiency. The ISP used in the iPhone XSmax is also evaluated as Apple experts deliberately calibrate it for outstanding quality. Note that the ISP is only required in the RGB-Vision framework.

Similarly, we use iPhone RAW images from the MultiRAW dataset in experiments. We have different ISP combinations for RGB-Vision to train and test RGB images (converted from the

same set of iPhone RAWs). The training and testing split is the same for either RGB-domain or RAW-domain processing.

As in Table S1 for the RGB-Vision category, the training ISP converts iPhone RAW images to the corresponding RGB samples to train YOLOv8-S, while the testing ISP is used to generate RGB samples (from iPhone RAW images) for testing previously trained YOLOv8-S.

The setup using the same iPhone ISP to generate RGB images for training and testing provides the best performance (see the last row of RGB-Vision in Table S1). Although we have tried our best to fine-tune the AX620A ISP to mimic the iPhone ISP, the setup using the same AX620A ISP (Tuned) to generate RGB images for training and testing is inferior to the case using the iPhone ISP that is deliberately calibrated by Apple imaging experts, e.g., 0.529 vs. 0.546 mAP. The detection performance is sharply degraded if we use different ISPs to generate training and testing RGB samples (see 1st and 2nd rows of Table S1 in RGB-Vision), suggesting that the ISP configuration is vital for task performance.

Fig. S2 visualizes detection results on testing images, further confirming the observations in Table S1 where inappropriate use of ISPs would lead to catastrophic performance degradation (see missing objects in the first column).

By contrast, under the ρ-Vision setup, YOLOv8-S is trained and tested on iPhone RAW images directly. The average detection performance is the same as using the iPhone ISP for both training and testing in RGB-vision. More importantly, expert tuning or dedicated calibration of ISP is no longer required. All of these suggest the encouraging prospects of using ρ-Vision in vision tasks.

Challenging Imaging Conditions are additionally examined to compare the efficiency of ρ-Vision and RGB-Vision pipelines. Two representative contexts are considered: the low-light illumination with high-noise levels and the scenario with high dynamic range (HDR) conditions.

Low-light illumination with high noise scenario is evaluated with object classification. We closely follow [*S1*] to perform the task, which involves training a MobileNet-V1 using noise-augmented ImageNet samples, then testing real-world noisy images acquired using a Google Pixel camera under low-light/high-noise conditions.

As for RGB-Vision, we directly train an RGB-domain MobileNet-V1 using the ImageNet dataset (RGB_{IN}) (with noise augmentation). In the meantime, we respectively use AX620A ISP and AX620A AI-ISP to transform RAW images acquired using Google Pixel camera (RAW_{GP}) to the corresponding RGB datasets, e.g., RGB_{AX} and $\text{RGB}_{\text{AX}-\text{AI}}$ to test aforementioned RGB-domain MobileNet-V1.

As for ρ-Vision, we first train our Unpaired CycleR2R model using clean RAW and RGB images from the Google Pixel and ImageNet datasets, i.e., RAW_{GP} and RGB_{IN}, respectively. Then, we use the invISP module in this Unpaired CycleR2R to convert RGB images in ImageNet to simulated RAW samples, i.e., $\text{simRAW}_{\text{IN}}$, to train the RAW-domain MobileNet-V1. The same noise augmentation is performed upon $\text{simRAW}_{\text{IN}}$. Such a RAW-domain MobileNet-V1 tests RAW samples directly from RAW_{GP}.

Evaluations presented in Table S2 clearly evidence the superiority of ρ-Vision paradigm. Notable reductions are reported for power consumption, memory footprint, and computational latency, owing to removing the ISP subsystem in the proposed ρ-Vision framework.

ρ-Vision only requires 0.006 J for task inference, compared to 0.128 and 0.162 J consumed by RGB-Vision methods using AX620A ISP and AX620A AI ISP. Furthermore, it exhibits the lowest latency at 2.71 ms, a substantial decrease from the 48.65 ms and 64.75 ms observed with the methods using AX620A ISP and AX620A AI ISP. This is because small-size images, e.g., 224×224, are used in the classifier, but ISPs must process images with the original resolution (2560×1440). Such a sharp increase in data volume increases power consumption, memory footprint, and latency.

ρ-Vision also presents better classification accuracy. We attribute it to noise separation and suppression in the RAW domain being more tractable than in the RGB domain (after a serial nonlinear transformation) [*S1*].

Notably, the AX620A AI ISP does not enhance classification performance under such extreme low-light conditions, as AX620A AI ISP models are typically trained for some specific cameras and may not generalize well to a new camera from the above discussions.

HDR conditions are studied with the detection task. 24-bit LUCID TRI054S RAW images (RAW_{LT}) covering the tunnel exit scenes are used. These HDR scenes are often encountered when driving through the tunnel and simultaneously experiencing extraordinarily bright and dark regions.

As for ρ-Vision, we train the RAW-domain detector (YOLOv8-S) using RAW_{LT}. In contrast, RAW samples in $\text{RAW}_{\text{LT}}$are first converted to RGB counterparts using the AX620A ISP to train the RGB-domain detector used in the RGB-Vision framework.

Besides the reductions in power consumption, memory footprint, and latency, the ρ-Vision framework achieves superior mAP across all categories, particularly in detecting traffic lights and signs (e.g., labeled as "Tr. L." and "Tr. S.") in Table S3. The improvement in mAP indicates the enhanced capability of the ρ-Vision to discern features in HDR conditions. This is essential for applications such as autonomous driving, where accurate and prompt traffic detection is crucial.

The combination of reduced latency, lower power consumption, and memory usage, along with higher mAP scores, affirms the effectiveness of the ρ-Vision framework in challenging HDR scenarios, highlighting its potential for real-world applications where both performance and efficiency are of paramount importance.

S.II. Details of the Unpaired CycleR2R

A. Architecture of Basic Neural Network

Table S4 details the architecture of the basic neural network $E(\cdot)$ used in Unpaired CycleR2R. This basic network $E(\cdot)$ consists of five layers in total and is used for IEM (Illumination Estimation Module), AWB (Auto White Balance), BA (Brightness Adjustment), and CC (Color Correction). The first layer applies the 5×5 convolution with 32 channels, and the

TABLE S2: **Classification Accuracy of RGB-Vision and ρ-Vision Frameworks Under Low-Light Conditions.** Latency measures the total processing duration by both the ISP and model, as well as the power consumption (Power.) and memory requirements (Mem.) for each method, besides the Top-1 classification accuracy (Acc.). *The results of Google Pixel ISP are copied from the paper [*S1*]. The "invISP" is used in ρ-Vision to generate simulated RAW samples to train the classifier, while RGB-vision methods do not require this step. RGB-Vision methods train the RGB-domain classifier using RGB images from the ImageNet dataset (RGB_{IN}) while ρ-Vision trains the RAW-domain classifier using simulated RAW images generated using the invISP. RAW images acquired by Google Pixel (RAW_{GP}) [*S1*] under extreme low-light conditions are used for evaluation. In the RGB-Vision pipeline, these RAW images are converted using different ISPs to RGB samples for using the RGB-domain classifier, while in the ρ-vision paradigm, these RAW images are directly fed to the RAW-domain classifier.

Method	**invISP**	**Classifier**		**Latency**		**Power.**	**Mem.**	**Acc.**
	Train	**Train**	**Test**	**ISP**	**Model**			
RGB-Vision w/ AX620A ISP	-	RGB_{IN}	RGB_{AX}	48.65 ms	2.73 ms	0.128 J	65 MB	0.0
RGB-Vision w/ AX620A AI-ISP	-	RGB_{IN}	$\text{RGB}_{\text{AX-AI}}$	64.75 ms	4.36 ms	0.162 J	81 MB	0.0
RGB-Vision w/ *Google Pixel ISP	-	RGB_{IN}	RGB_{GP}	-	-	-	-	1.4
ρ-Vision	RGB_{IN}, RAW_{IN}	$\text{simRAW}_{\text{IN}}$	RAW_{GP}	**0 ms**	**2.71 ms**	**0.006 J**	**25 MB**	**19.8**

TABLE S3: **Comparative Analysis of RGB-Vision and ρ-Vision Frameworks in High Dynamic Range (HDR) Scenarios.** The RAW-domain detector is calibrated with 24-bit LUCID TRI054S RAW images (RAW_{LT}). The RGB-domain detector is trained and evaluated on RGB images generated using AX620A ISP (RGB_{AX}). Latency encompasses the total processing time of both the ISP and the detection model. We present the power consumption (Power.) and memory footprint (Mem.) alongside the mean Average Precision (mAP). Abbreviations "Tr. L." and "Tr. S.", denote traffic light and traffic sign, respectively.

Framework	**Detector**		**Latency**		**Power.**	**Mem.**	$\textbf{AP}_{\textbf{Car}}$	$\textbf{AP}_{\textbf{Tr. L}}$	$\textbf{AP}_{\textbf{Tr. S}}$	**mAP**
	Train	**Test**	**ISP**	**Model**						
RGB-Vision	RGB_{AX}	RGB_{AX}	48.55 ms	**17.07 ms**	0.152 J	55 MB	81.3	27.9	61.2	56.8
ρ-Vision	RAW_{LT}	RAW_{LT}	**0 ms**	18.18 ms	**0.058 J**	**35 MB**	**84.8**	**35.5**	**69.7**	**63.3**

subsequent two layers use 3×3 convolutions and 64 channels. The final two layers use simple linear layers instead.

The example of "Conv: k5c32s2" stands for a convolutional layer having convolutions with spatial kernel size at 5×5 (k5), 32 channels (c32), and a stride of two based spatial downsampling (s2) at both dimensions. The same convention is applied to the linear layer (Linear) and average pooling layer (Avg Pool). "Leaky RELU" [*S4*] is used as the activation, and "Mean" stands for the average operator in the spatial domain for each channel. Considering the output channel of $E(\cdot)$ is specific for different purposes across aforementioned modular components, we mark it using a predefined variable C_{out}.

B. Architectures of Discriminators

As in the main paper, D_{color} and D_{bright} are applied to measure the similarity between generated and real images. D_{color} stacks five convolutional layers with Leaky ReLU [*S4*] and D_{bright} uses five linear layers instead to process 1D grayscale histogram. Details of kernel size, channels, and strides are listed in Tabel. S4.

C. Gamma Correction Standard

Gamma correction matches the non-linear characteristics of a display device or human perception [*S3*]. We adopt the correction function recommended in ITU-R BT. 709 standard [*S4*], noted as f_g, which is widely used in commodity ISPs today [*S4*].

$$\begin{aligned}\boldsymbol{y} &= f_g \circ \boldsymbol{x}_{cc} \\ &= \begin{cases} 12.92 \cdot \boldsymbol{x}_{cc}, & \boldsymbol{x}_{cc} \leq 0.00304, \\ 1.055 \cdot \boldsymbol{x}_{cc}^{1/2.4} - 0.055, & \boldsymbol{x}_{cc} > 0.00304. \end{cases}\end{aligned} \tag{S1}$$

Correspondingly, the inverse function g_g is:

$$\begin{aligned}\boldsymbol{x}_{cc} &= g_g \circ \boldsymbol{x}_g \\ &= \begin{cases} \dfrac{\boldsymbol{y}}{12.92}, & \boldsymbol{y} \leq 0.04045, \\ \left(\dfrac{\boldsymbol{y} + 0.055}{1.055}\right)^{2.4}, & \boldsymbol{y} > 0.04045. \end{cases}\end{aligned} \tag{S2}$$

S.III. Details of *Distribution Analysis of RAW images*

A. The proof of the equation (28)

We start from the loss function $\mathcal{L}$:

$$\mathcal{L} = \frac{1}{H \times W} \left(\boldsymbol{w} * (\boldsymbol{P} - 0.5) + \mathbf{b} - \hat{\mathbf{y}}\right)^2, \tag{S3}$$

where $\boldsymbol{w} \in \mathbb{R}^{S \times S}$ is the convolution kernel with kernel size S.

Then the partial derivative of $w \in \boldsymbol{w}$ could be formulated as:

$$\frac{\partial \mathcal{L}}{\partial w} = \frac{1}{H \times W} \sum_{j=0}^{H} \sum_{i=0}^{W} 2 \left(\mathbf{y}_{ij} - \hat{\mathbf{y}}\right) \left(\mathbf{x}_{ij+mn} - 0.5\right) \tag{S4}$$

where $\mathbf{x}_{ij+mn} \in \boldsymbol{P}$ and mn is the shift position of w according to the kernel center of $\boldsymbol{w}$. $\mathbf{y}_{ij}$ is the convolution output at position ij. To calculate $\mathbf{y}_{ij}$, we define $\boldsymbol{x}_{ij}^{w}$ as a window of $\boldsymbol{P}$ with the same size of $\boldsymbol{w}$ located at ij. Considering the similarity among adjacent pixels, for a neighborhood pixel of $\mathbf{x}_{ij}$, *i.e.*, $\mathbf{x}_{neibor} \in \boldsymbol{x}_{ij}^{w}$, we have $\mathbf{x}_{neibor} = \mathbf{x}_{ij} + \delta$, where δ follows a Gaussian distribution with zero mean. Thus, $\mathbf{y}_{ij}$ could be expanded as:

$$\mathbf{y}_{ij} = (\mathbf{x}_{ij} - 0.5) \sum w + + \sum w\delta. \tag{S5}$$

For simplify, we use $\tilde{\boldsymbol{x}}$ and $\tilde{\mu}$ to replace $\boldsymbol{x} - 0.5$ and $\mu - 0.5$, respectively. Besides, we set $A = \sum w,\ B = + \sum w\delta,\ C = 2A,\ D = 2\left(B - \hat{\mathbf{y}}\right)$. Having $\mathbf{y}_{ij} = A\tilde{\mathbf{x}} + B$, the $\frac{\partial \mathcal{L}}{\partial w}$ will be:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial w} &= 2\mathbb{E}\left[\left(\mathbf{y} - \hat{\mathbf{y}}\right)(\mathbf{x} - 0.5)\right] \\
&= 2\mathbb{E}\left[\left(A\tilde{\mathbf{x}} + B - \hat{\mathbf{y}}\right)\tilde{\mathbf{x}}\right] \\
&= 2\mathbb{E}\left[A\tilde{\mathbf{x}}^2 + (B - \hat{\mathbf{y}})\tilde{\mathbf{x}}\right] \\
&= 2\mathbb{E}\left[A\right]\mathbb{E}\left[\tilde{\mathbf{x}}^2\right] + 2(\mathbb{E}\left[B\right] - \hat{\mathbf{y}})\mathbb{E}\left[\tilde{\mathbf{x}}\right] \\
&= 2A(\tilde{\mu}^2 - \sigma^2) + 2(b - \hat{\mathbf{y}})\tilde{\mu} \\
&= C(\tilde{\mu}^2 - \sigma^2) + D\tilde{\mu}.
\end{aligned} \tag{S6}$$

Since μ and σ are independent and we only concern with the impact of $p\left(\mu\right)$, we set Var $\left[\sigma^2\right]$ to a constant. Then the variance could be expanded as:

$$\begin{aligned}
\text{Var}\left[\frac{\partial \mathcal{L}}{\partial w}\right] &= \text{Var}\left[C\tilde{\mu}^2 + D\tilde{\mu}\right] + \text{Var}\left[C\sigma^2\right] \\
&= \mathbb{E}\left[\left(C\tilde{\mu}^2 + D\tilde{\mu}\right)^2\right] - \mathbb{E}\left[C\tilde{\mu}^2 + D\tilde{\mu}\right]^2 \\
&\quad + \text{const} \\
&= \mathbb{E}\left[C^2\tilde{\mu}^4\right] + \cancel{\mathbb{E}\left[CD\tilde{\mu}^3\right]} + \mathbb{E}\left[D^2\tilde{\mu}^2\right] \\
&\quad - \left(\mathbb{E}\left[C\tilde{\mu}^2\right] + \cancel{\mathbb{E}[D\tilde{\mu}]}\right)^2 + \text{const} \\
&= \mathbb{E}\left[\left(C\tilde{\mu}^2\right)^2\right] - \left(\mathbb{E}\left[C\tilde{\mu}^2\right]\right)^2 \\
&\quad + \mathbb{E}\left[\left(D\tilde{\mu}\right)^2\right] - \left(\mathbb{E}\left[D\tilde{\mu}\right]\right)^2 + \text{const} \\
&= \text{Var}\left[C\tilde{\mu}^2\right] + \text{Var}\left[D\tilde{\mu}\right] + \text{const} \\
&= C^2\text{Var}\left[\tilde{\mu}^2\right] + D^2\text{Var}\left[\tilde{\mu}\right] + \text{const}.
\end{aligned} \tag{S7}$$

TABLE S4: Network settings of Unpaired CycleR2R.

Basic Encoder $E\left(\cdot\right)$	Discriminator D_{color}	Discriminator D_{bright}
Conv: k5c32s2	Conv: k4c64s2	Linear: c1024
Leaky RELU	Leaky RELU	Leaky RELU
Avg Pool: s2	Conv: k4c128s2	Linear: c1024
Conv: k3c64s2	Leaky RELU	Leaky RELU
Leaky RELU	Conv: k4c256s2	Linear: c256
Avg Pool: s2	Leaky RELU	Leaky RELU
Conv: k3c64s1	Conv: k4c512s2	Linear: c256
Leaky RELU	Leaky RELU	Leaky RELU
Mean	Conv: k4c1s2	Linear: c1
Linear: c256	Mean	-
Linear: cC_{out}	-	-

B. *The proof of the equation* (29)

Given the μ following the distribution in (25), the Var $[\tilde{\mu}]$ could be written as:

$$\begin{aligned}
\text{Var}\left[\tilde{\mu}\right] &= \text{Var}\left[\mu - 0.5\right] = \text{Var}\left[\mu\right] \\
&= \int_0^1 \left[\mu - \mathbb{E}\left(\mu\right)\right]^2 p\left(\mu\right) \mathrm{d}\mu \\
&= \int_0^1 \left(\mu - 0.5\right)^2 \left(k\mu^2 - k\mu + \frac{k}{6} + 1\right)\mathrm{d}\mu \\
&= F(\mu = 1) - F(\mu = 0) \\
&= \left(\frac{1}{21} - \frac{k}{720}\right) - \left(-\frac{k}{144} - \frac{1}{24}\right) \\
&= \frac{k}{180} + \frac{1}{12},
\end{aligned} \tag{S8}$$

where $F\left(\mu\right) = k\left(\frac{\mu^5}{5} - \frac{\mu^4}{2} + \frac{\mu^3}{12} - \frac{\mu^2}{8}\right) + \frac{k}{18}\left(\mu - \frac{1}{2}\right)^3$.

Thus, Var $\left[\frac{\partial \mathcal{L}}{\partial w}\right]$ will be:

$$\begin{aligned}
\text{Var}\left[\frac{\partial \mathcal{L}}{\partial w}\right] &\approx D^2\text{Var}\left[\tilde{\mu}\right] + \text{const} \\
&= D^2\left(\frac{k}{180} + \frac{1}{12}\right) + \text{const} \\
&= D^2\frac{k}{180} + \text{const}.
\end{aligned} \tag{S9}$$

S.IV. RAW-DOMAIN CLASSIFICATION

In this section, we present the application of our Unpaired CycleR2R model for the classification task in the RAW domain.

A. *Datasets and Baselines*

We utilize the identical dataset for training and testing as in [*S1*]. For generating the training set, we use ImageNet [*S3*] to generate simulated RAW images with noises. As for testing, a real-world RAW dataset captured by a Google Pixel camera, e.g., RAW_{GP}, is used. This dataset collects images acquired with low-light conditions spanning a range of illumination from 1 lux to 200 lux and containing 1103 images in 40 categories.

We employed the MobileNet-V1 [*S4*] for classification as suggested by [*S1*].

As for the proposed ρ-Vision, Unpaired CycleR2R is first trained using RGB images in ImageNet (RGB_{IN}) and Google Pixel RAWs (RAW_{GP}) to generate a simulated RAW dataset ($\text{simRAW}_{\text{IN}}$). This $\text{simRAW}_{\text{IN}}$ is augmented with noises and applied to train the RAW-domain classifier MobileNet-V1. Consequently, the trained RAW-domain MobileNet-V1 examines the testing RAWs from RAW_{GP} for task inference.

As for the Anscombe ISP method proposed in [*S1*], ImageNet RGB images (RGB_{IN}) undergo mosaic operations to generate simulated RAWs, which are then injected with Gaussian-Poisson noise to produce noisy simRAWs. The training has two steps: First, the Anscombe ISP is trained with paired noisy RAW and clean RGB images. Second, Anscombe ISP and Imagenet pre-trained MoblieNet-V1 are jointly trained using noisy simRAWs and classification label annotations. During the testing, the Anscombe ISP converts Google Pixel

TABLE S5: **Classification Accuracy On Google Pixel RAW images.**

Method	invISP	Classifier		Top-1 Acc.	Top-5 Acc.	# Parameters	FLOPs
	Train	Train	Test				
Anscombe ISP*[*S1*]	-	$RGB_{Ans\text{-}ISP}$	$RGB_{Ans\text{-}ISP}$	33.1	58.4	4.28	282
Mosaic RAW*[*S1*]	-	$simRAW_{IN}$	RAW_{GP}	27.0	52.5	**4.23**	**181**
Unpaired CycleR2R	RGB_{IN}, RAW_{GP}	$simRAW_{IN}$	RAW_{GP}	**35.5**	**72.1**	**4.23**	**181**

* Both the Anscombe ISP and Mosaic RAW apply simple mosaic operations to generate RAW samples from the corresponding RGB images. They don't need to train the invISP.

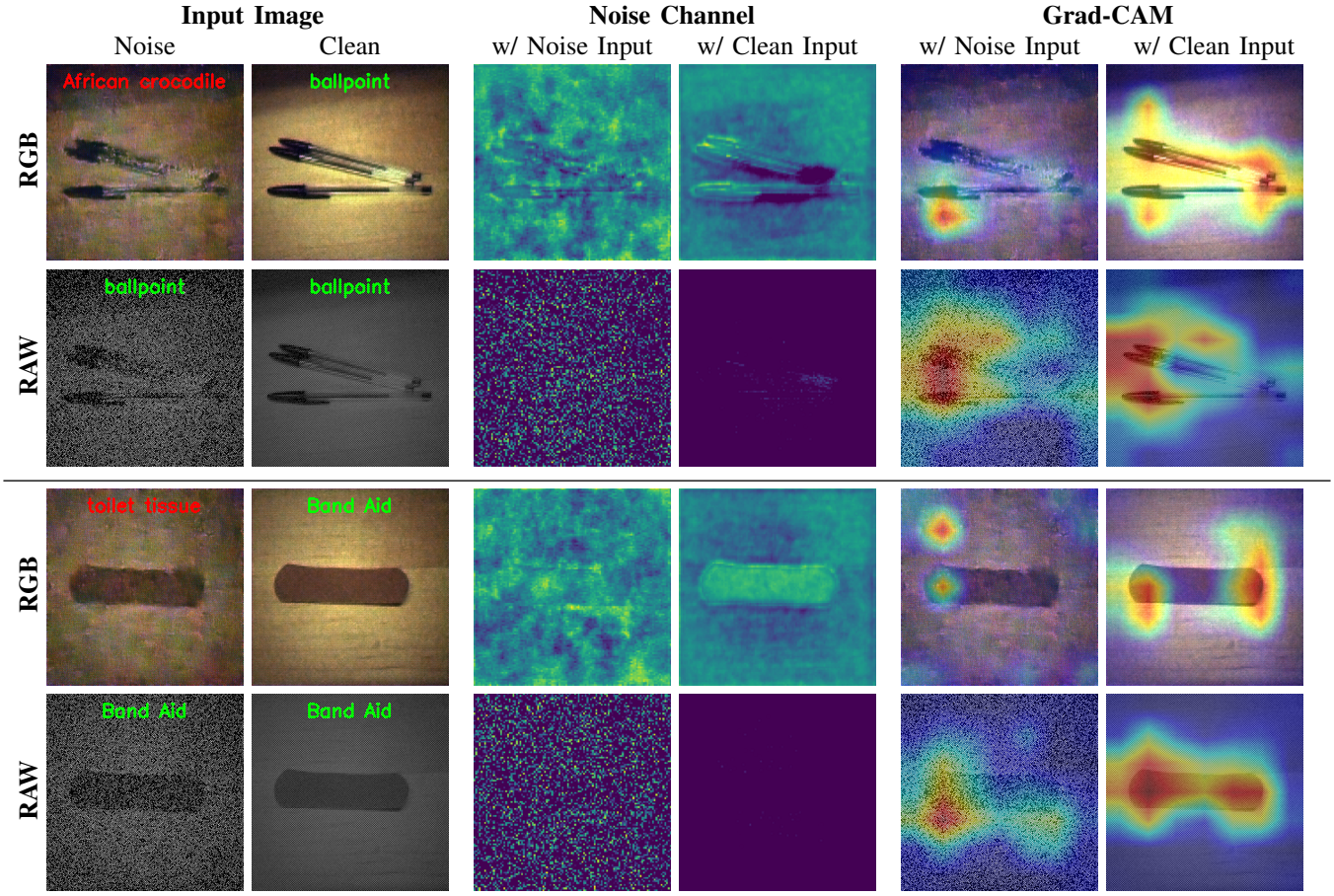


Fig. S3: **Visualization of Classifier Response to Noisy and Clean Inputs** The "RGB" rows represent the processing using the Anscombe ISP [*S1*] where it inputs the RGB image for classification; In contrast, the "RAW" rows stand for the processing using Unpaired CycleR2R where the RAW images are directly processed. Noise is augmented upon the clean inputs to form Noisy samples. The "Noise Channel" is the feature channel in the shallow layer "Conv2d_0" that presents the maximum difference when processing the noise and clean inputs respectively. The Grad-CAM [*S2*] visualizations are based on the last convolutional layer "Conv2d_13_pointwise". A comparison between the "Noise Channel" under different inputs reveals that the RAW-domain classifier is adept at extracting noise patterns, effectively separating noise from the signal, which results in Grad-CAM visualizations that more closely resemble the clean input. In contrast, the RGB-domain classifier struggles to disentangle noise from the signal due to the complex non-linear processing by the Anscombe ISP, leading to significant deviations in Grad-CAM under noisy conditions and consequently to misclassification.

RAW images (RAW_{GP}) to the corresponding RGB format ($\text{RGB}_{\text{Ans-ISP}}$) for classification.

As for the Mosaic RAW method [*S1*], ImageNet images are simply mosaiced to drive RAW samples to form the $\text{simRAW}_{\text{IN}}$. Noise is then augmented onto the $\text{simRAW}_{\text{IN}}$ to train the RAW-domain classifier. Subsequently, samples in (RAW_{GP}) are tested directly.

Note that noise augmentation closely follows the studies in [*S1*] for all approaches.

B. Comparative Studies of RAW-domain Classification

Table S5 reports the image classification under low-light illumination with high noises. The proposed $\rho-Vision$ using Unpaired CycleR2R demonstrates the compellingly superior performance to the approaches, e.g., Anscombe ISP and Mosaic RAW, provided by [*S1*].

The gain of the proposed Unpaired CycleR2R to the Mosaic RAW owes the better characterization of real-life RAW images in training/devising the invISP to generate realistic simRAWs. The Mosaic RAW approach [*S1*], instead, only applies the basic mosaicking by simply neglecting the impacts of gamma correction and white balance that are vital in the transformation between RGB and RAW space..

The improvement of the Anscombe ISP to the Mosai RAW is due to the mapping between a noisy RAW image and the corresponding clean RGB sample offered by the Anscombe ISP, which significantly helps the subsequent task.

The gain of the proposed Unpaired CycleR2R to the Anscombe ISP is attributed to the better noise separation and suppression in the RAW domain. This improvement is visually corroborated in Fig. S3, where the "Noise Channel" columns under the Unpaired CycleR2R method (RAW row) exhibit a more apparent distinction between noisy and clean features. The efficacy of our model in noise modeling and separation in the RAW domain, as proofed in [*S1*], is further evidenced by the Grad-CAM visualizations. These visualizations of noisy inputs are similar to those generated from clean inputs, illustrating the model's ability to preserve essential image characteristics despite noise. In contrast, the Anscombe ISP (RGB row) reveals a significant disparity in the Grad-CAM outputs when comparing noisy and clean inputs, which may lead to classification errors.

Our Unpaired CycleR2R achieves this superior noise discrimination without increasing computational complexity, thereby maintaining the same level of FLOPs as the Mosaic RAW (lower than the Anscombe ISP).

S.V. RAW-domain Segmentation

In the main text of this paper, the detection task is successfully executed in the RAW domain with superior performance to that using the same RGB-domain model. Here we explore the feasibility of RAW-domain segmentation. Similar to the discussions in Sec. 5.2 and 6.2, we first demonstrate that the segmentation model trained with simRAW images can directly infer the segmentation cues upon the real RAW images. Second, a few-shot finetuning simRAW-pretrained segmentation model using limited labeled real RAW images further improves its performance and shows consistent gains to the model trained from scratch. Finally, ablation studies show that gamma correction is also vital for segmentation tasks in the RAW domain.

A. Datasets

Cityscapes [*S5*] is a large-scale dataset recorded in different urban streets in Europe containing 5,000 frames with high-quality pixel-level segmentation annotations. Considering the different traffic signs in China where the MultiRAW is captured, we use a communal subset including road, building, fence, traffic light, sky, person, car, truck, and bus for evaluation. Following the setup in Sec. 5.2 of the main paper, we convert the RGB samples, a.k.a RGB_{c}, into simRAW image set simRAW_{c} to train/refine RAW-domain segmentation model.

B. Training Details

We use the famous HRNetv2 [*S5*] as our segmentation network. All segmentation models are optimized by a SGD optimizer with 0.9 momentum, 5×10^{-4} weight decay and initial learning rate of 10^{-2} dropped into 10^{-4} linearly. The batch size is set as 8, and inputs are randomly cropped into 512×1024 with random flip augmentation. The experiments are conducted using an Nvidia 3090Ti GPU.

C. Comparative Studies of RAW-domain Segmentation

Table S6 and Fig. S4 compares our Unpaired CycleR2R and other methods using invISP approach [*S1*, *S1*, *S1*, *S2*] and domain-adaptation (DA) solution [*S6*, *S7*]. It can be seen that Unpaired CycleR2R outperforms the state-of-the-art CycleISP by a significant margin of $7\,\text{mIoU}$ and improves the IoU across all classes of objects. More gains are presented against other approaches.

Our model also surpasses the RGB Baseline on mIoU. Note that this RGB Baseline is prevalent in real-world applications. Such a convincing performance suggests the potential for RAW-domain segmentation. We also observe the lower IoU for some specific classes of objects between our method and the RGB Baseline. This is probably due to the optimization strategy for maximizing the overall performance but not balancing each class. This is an interesting topic for future study.

Apparently, inputting real RAW images to the RGB-domain segmentation model directly for task execution is a failure, as exemplified in the Naive Baseline, e.g., mIoU of 11.1 versus the mIoU of 47.5 in the RGB Baseline, which is due to the large discrepancy between the RGB-domain and RAW-domain models.

Implementation Friendliness. As aforementioned in Sec. 5.2, our method could generate simRAW images to train task-dependent models. However, DA-based approaches [*S1*, *S1*] designed for object detection tasks could not be applied to segmentation tasks. And DA-based segmentation methods [*S6*, *S7*] could not support the detection task either.

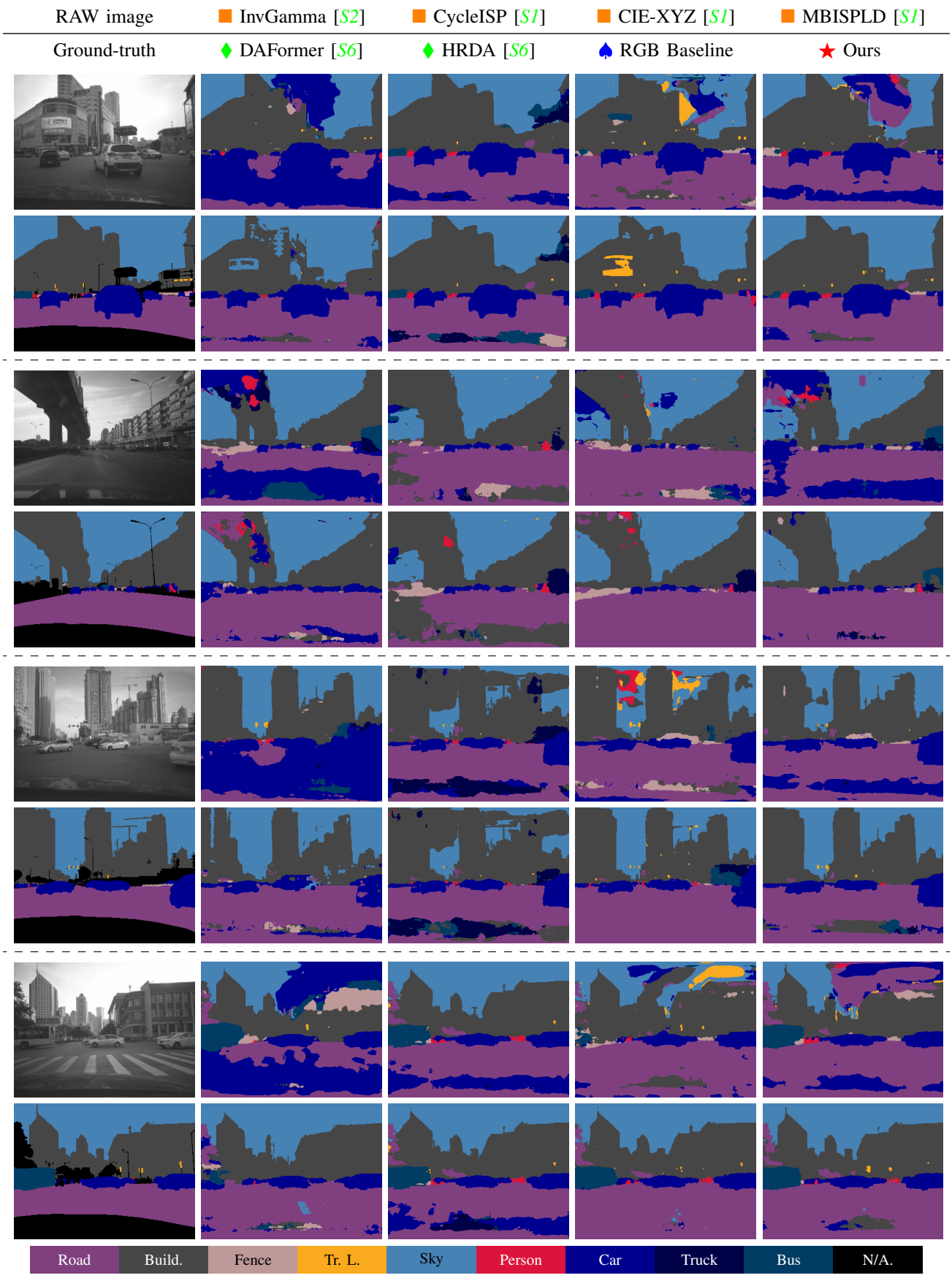


Fig. S4: **Qualitative Visualization of Pretrained RAW Segmentation Model.** Example predictions show better recognition of buildings, sky, and traffic lights by our Unpaired CycleR2R on Cityscapes RGB → iPhone RAW. Gamma correction and brightness adjustment have been applied to RAW images for a better view.

TABLE S6: **mIoU (Mean Intersection over Union) of Segmentation on the testing set of iPhone RAW images.** RGB-domain segmentation model is trained using original RGB images in *Cityscapes* [*S5*] (e.g., RGB_c); Various simRAW datasets associated with RGB_c are generated using different methods which are marked as $simRAW_c$ to train RAW-domain segmentation model. The testing RAW images in iPhone RAW RAW_i and their paired RGB images in RGB_i converted using built-in iPhone ISP are tested accordingly. HRNetv2 [*S5*] is used as the base segmentation model. ♠ Baselines, ♦ Domain Adaptation Solutions, ■ invISP Methods, ★ Ours.

Method	invISP	Segmentor		Road	Build.	Fence	Tr. L.	Sky	Person	Car	Truck	Bus	mIoU
	Train	Train	Test										
♠ Naive Baseline	-	RGB_c	RAW_i	0.3	21.6	14.8	5.7	20.7	0.4	30.0	0.4	6.2	11.1
♠ RGB Baseline	-	RGB_c	RGB_i	**89.6**	65.1	35.6	20.7	**96.1**	11.1	62.9	**21.5**	25.3	47.5
♦ DAFormer (CVPR'22) [*S6*]	-	RGB_c, RAW_i	RAW_i	75.8	49.5	15.2	1.5	90.0	5.3	58.3	0.2	6.3	32.9
♦ HRDA (ECCV'22) [*S7*]	-	RGB_c, RAW_i	RAW_i	73.8	69.1	38.5	12.3	80.6	15.0	51.2	16.2	20.9	42.0
■ InvGamma (ICIP'19) [*S2*]	RGB_i, RAW_i	$simRAW_c$	RAW_i	47.5	55.7	31.2	8.3	90.0	7.3	23.9	11.2	17.6	32.5
■ CycleISP (CVPR'20) [*S1*]	RGB_i, RAW_i	$simRAW_c$	RAW_i	84.8	63.9	35.0	18.0	86.3	9.7	55.7	18.0	20.6	43.6
■ CIE-XYZ Net (TPAMI'21) [*S1*]	RGB_i, RAW_i	$simRAW_c$	RAW_i	78.7	64.4	36.7	3.0	84.2	5.4	48.6	2.3	15.4	37.6
■ MBISPLD (AAAI'22) [*S1*]	RGB_i, RAW_i	$simRAW_c$	RAW_i	72.5	60.8	39.4	7.3	78.6	13.3	41.0	17.7	20.8	39.0
★ **Unpaired CycleR2R**	RGB_c, RAW_i	$simRAW_c$	RAW_i	88.9	**70.5**	**40.9**	**24.7**	95.5	**21.4**	**64.3**	19.1	**30.0**	**50.6**

Build. ← Building; **Tr. L.** ← Traffic Light.

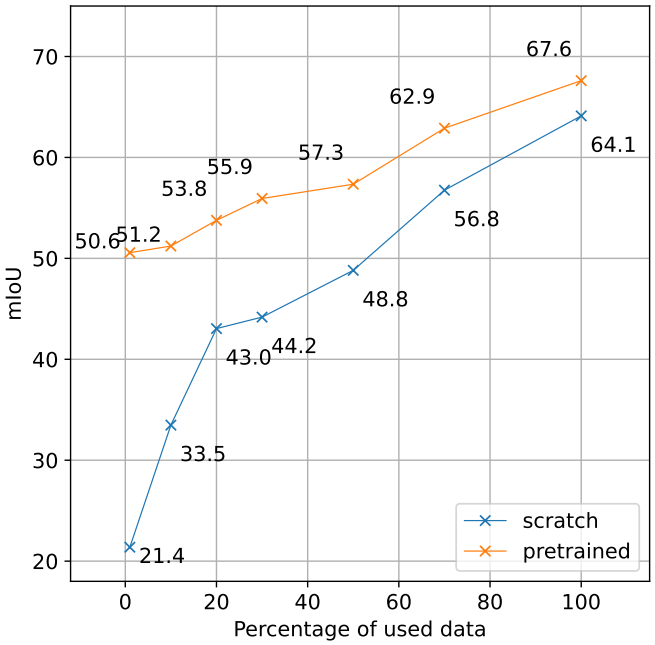


Fig. S5: **Few-shot finetuning using limited camera RAWs.** The simRAW-pretrained HRNetv2 [*S5*] is obtained by using samples in $simRAW_c$ generated by our Unpaired CycleR2R, which is then finetuned using limited camera RAW images; and the "scratch" model is randomly initialized and then trained using the same number of labeled real RAW images.

D. Comparative Studies of Few-Shot Finetuning

The performance of the simRAW-pretrained segmentation model could be further boosted by feeding more real labeled RAW images. We further finetune our segmentation model using our MultiRAW dataset (iPhone XSmax) with all classes. As depicted in Fig. S5, the segmentation accuracy is improved and consistently outperforms the scratch model which is initialized randomly and then trained using the same labeled real RAW images.

S.VI. Extra Quantitative Visualization

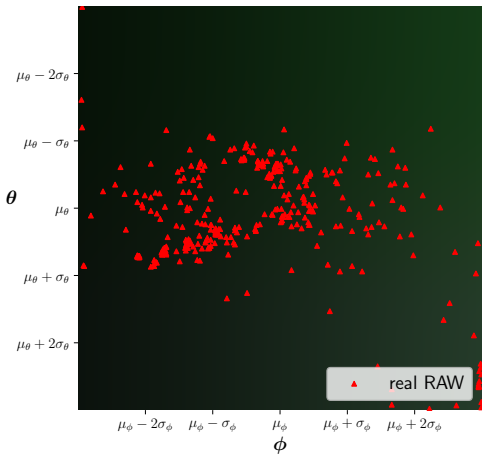
In Fig. S6, we present a visual comparison between our simulated RAW images and real RAW images. We also offer more qualitative visualizations of our lossy RIC at low Bits-rate and high Bits-rate in Fig. S7 and Fig. S8 respectively. Similar to the results in the main content of this work, we can clearly observe the subjective improvements of the proposed lossy RIC compared to the HEVC and VVC. Especially for the traffic light and car information, our lossy RIC provides sharper and less noisy reconstructions closer to the ground truth samples. Also, we give visualizations of progressive decoding using our lossless RIC within various cameras in Fig. S9-S11. Our lossless RIC could provide low-resolution previews for different cameras (iPhone XSmax, Huawei P30pro, and asi 294mcpro) and different scenes (both daylight and nighttime), which is helpful for professional photography and network transmission.

References

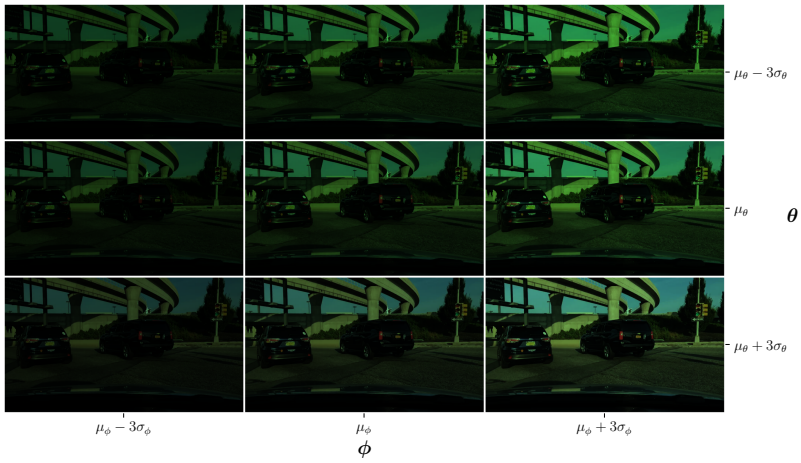
[*S1*] S. Diamond, V. Sitzmann, F. Julca-Aguilar, S. Boyd, G. Wetzstein, and F. Heide, "Dirty pixels: Towards end-to-end image processing and perception," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 3, pp. 1–15, 2021. 3, 4, 5, 6, 7

[*S2*] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626. 6

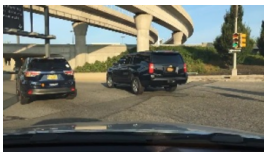
[*S3*] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255. 5



(a) simRAW is a superset of real RAW data



(b) simRAW examples



(c) Testing RGB

(d) Random real RAW images from multiRAW dataset

(e) simRAW w/o IEM

(f) simulating RAW images by injecting noise to $E^{-1}(\cdot)$ outputs w/o using IEM

Fig. S6: **Evaluation of the Illumination Estimation Module (IEM).** *Demosaicing has been applied to all images to enhance visibility.* (a) Adapting IEM to generate the simRAW's coverage using the mean color, where the color of each point ϕ_i, θ_j represents the average color of a simRAW generated by sampled illumination parameters ϕ_i, θ_j. In contrast, red markers indicate the average color of real RAW images. It clearly reveals that adapting IEM can cover all real-world illumination conditions in real RAW data. (b) simRAW examples generated by our Unpaired CycleR2R with various ϕ, θ, illustrating the IEM's ability to produce a wide range of illumination variations. (c) The corresponding RGB image fed into the invISP of our Unpaired CycleR2R, which is from the BDD100K dataset. (d) Random real RAW images from the multiRAW dataset, displaying the natural variability in illumination and color temperature. (e) Simulating a RAW image without using IEM, which can only produce a single simRAW per RGB input due to the absence of probabilistic illumination estimation. (f) Simulating RAW images by injecting noise to $E^{-1}(\cdot)$ outputs, which can produce multiple simRAW samples without requiring IEM but demonstrate unrealistic diversity induction in RAW Images. $E^{-1}(\cdot)$ is defined in invISP (see Fig. 2 in the main paper).

[*S4*] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017. 5

[*S5*] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7, 9

[*S6*] L. Hoyer, D. Dai, and L. Van Gool, "Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9924–9935. 7, 8, 9

[*S7*] ——, "Hrda: Context-aware high-resolution domain-adaptive semantic segmentation," *arXiv preprint arXiv:2204.13132*, 2022. 7, 9

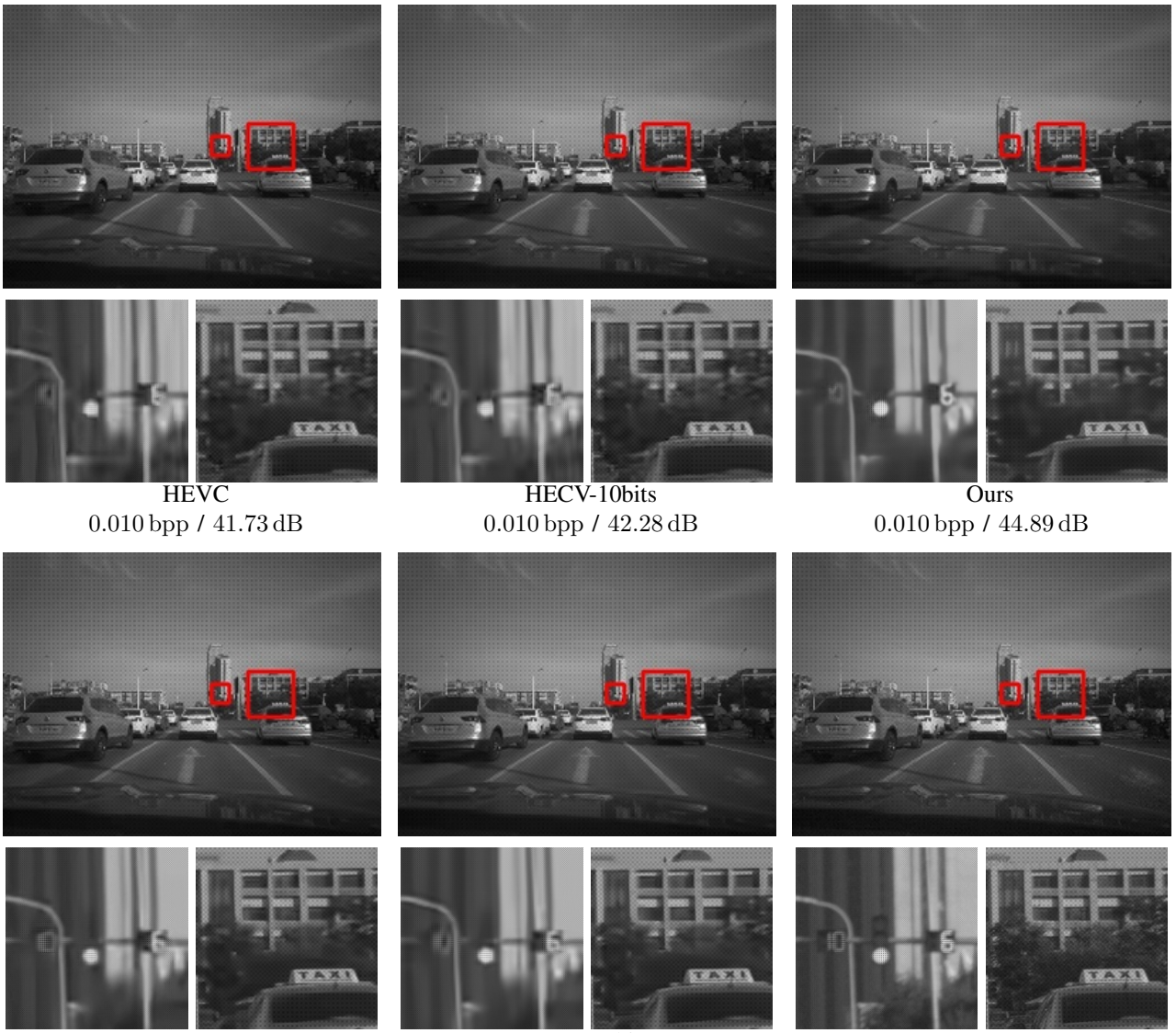


Fig. S7: **Qualitative Visualization of Lossy RIC at Low Bits-rate.** Reconstructions and close-ups of the HEVC, VVC, and our method. Corresponding bpp and PSNR are marked. Gamma correction and brightness adjustment have been applied for a better view. *Zoom for better details.*

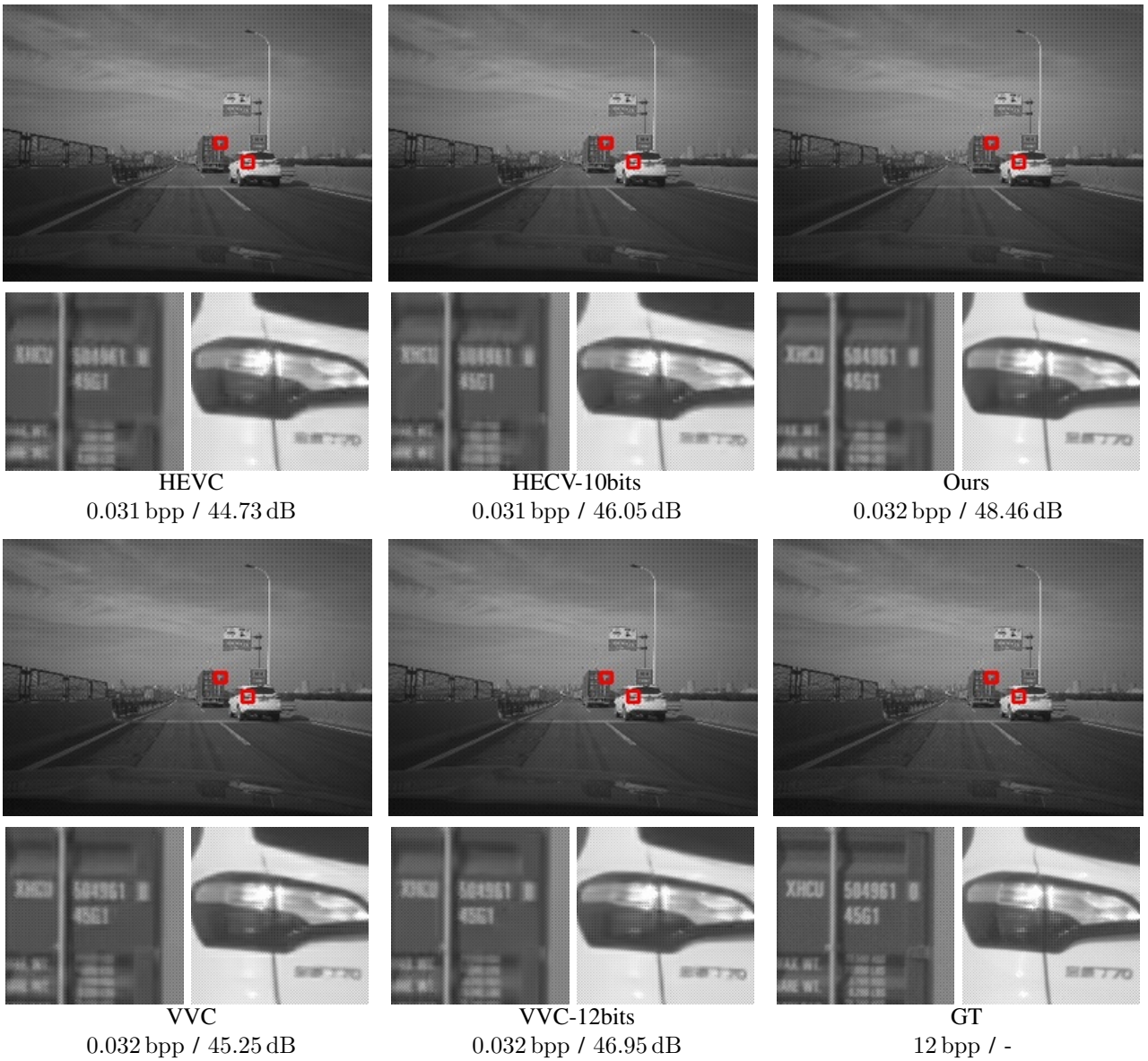


Fig. S8: **Qualitative Visualization of Lossy RIC at High Bits-rate.** Reconstructions and close-ups of the HEVC, VVC, and our method. Corresponding bpp and PSNR are marked. Gamma correction and brightness adjustment have been applied for a better view. *Zoom for better details.*

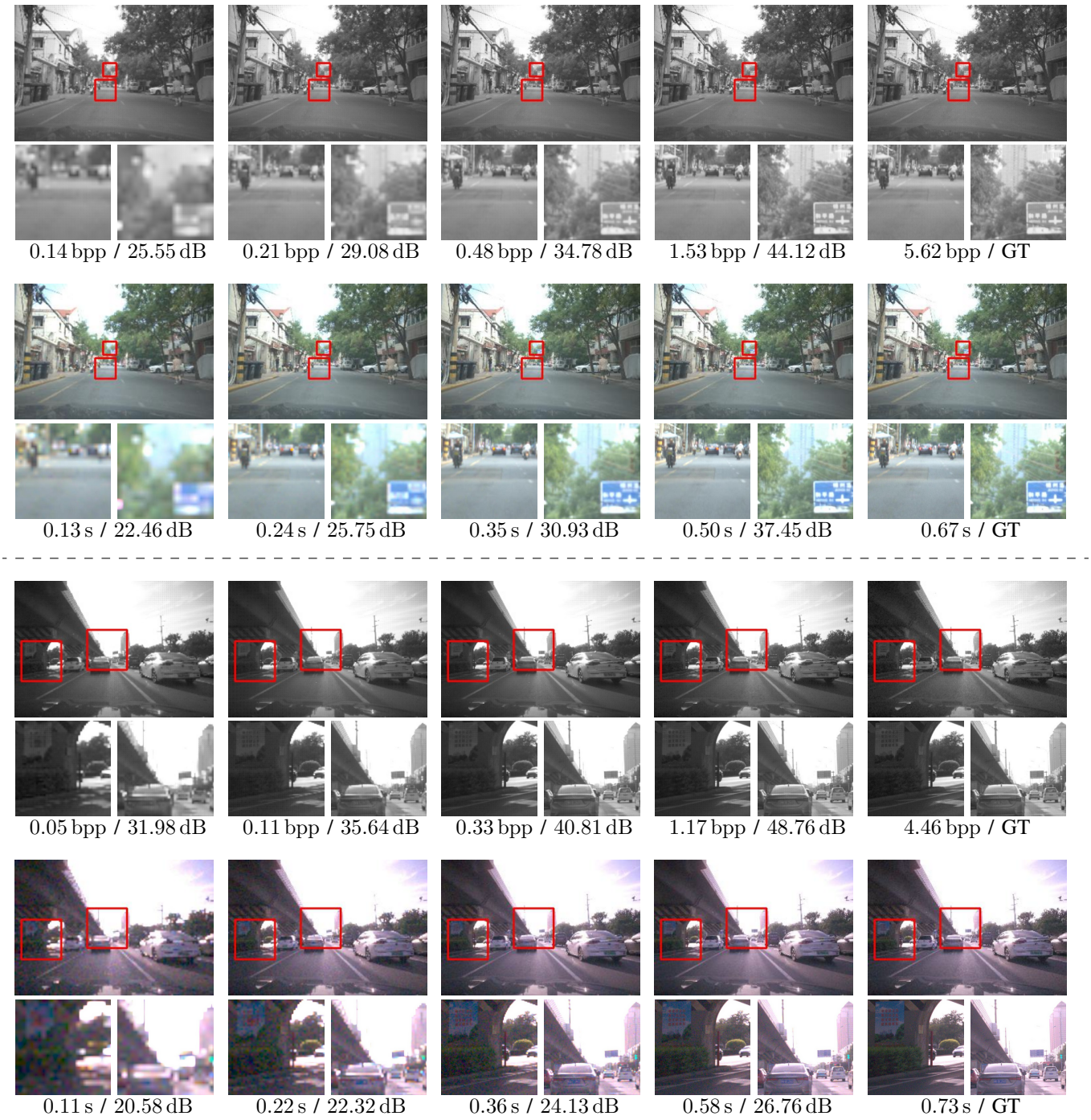


Fig. S9: **Qualitative Visualization of Lossless RIC Progressive Decoding (iPhone XSmax).** The gradual reconstruction of RAW images and their corresponding RGB images converted by an in-camera ISP. Bits per pixel (bpp) / PSNR (dB) is shown under RAW images. Decoding latency (s) / PSNR (dB) is also listed below RGB images. PSNR is derived against the GT (ground truth). Gamma correction and brightness adjustment have been applied for a better view. *Zoom for more details.*

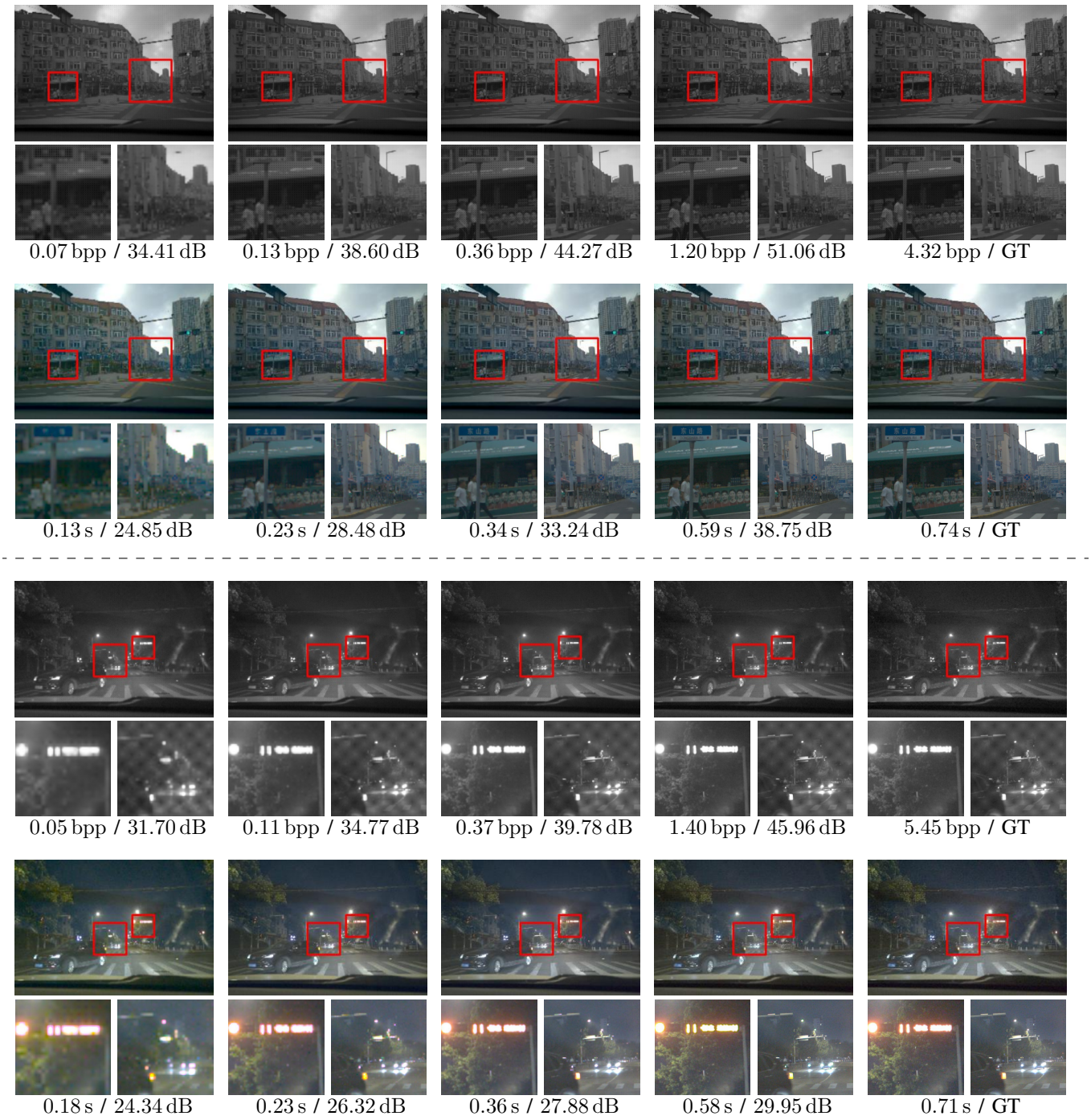


Fig. S10: **Qualitative Visualization of Lossless RIC Progressive Decoding (Huawei P30pro).** The gradual reconstruction of RAW images and their corresponding RGB images converted by an in-camera ISP. Bits per pixel (bpp) / PSNR (dB) is shown under RAW images. Decoding latency (s) / PSNR (dB) is also listed below RGB images. PSNR is derived against the GT (ground truth). Gamma correction and brightness adjustment have been applied for a better view. *Zoom for more details.*

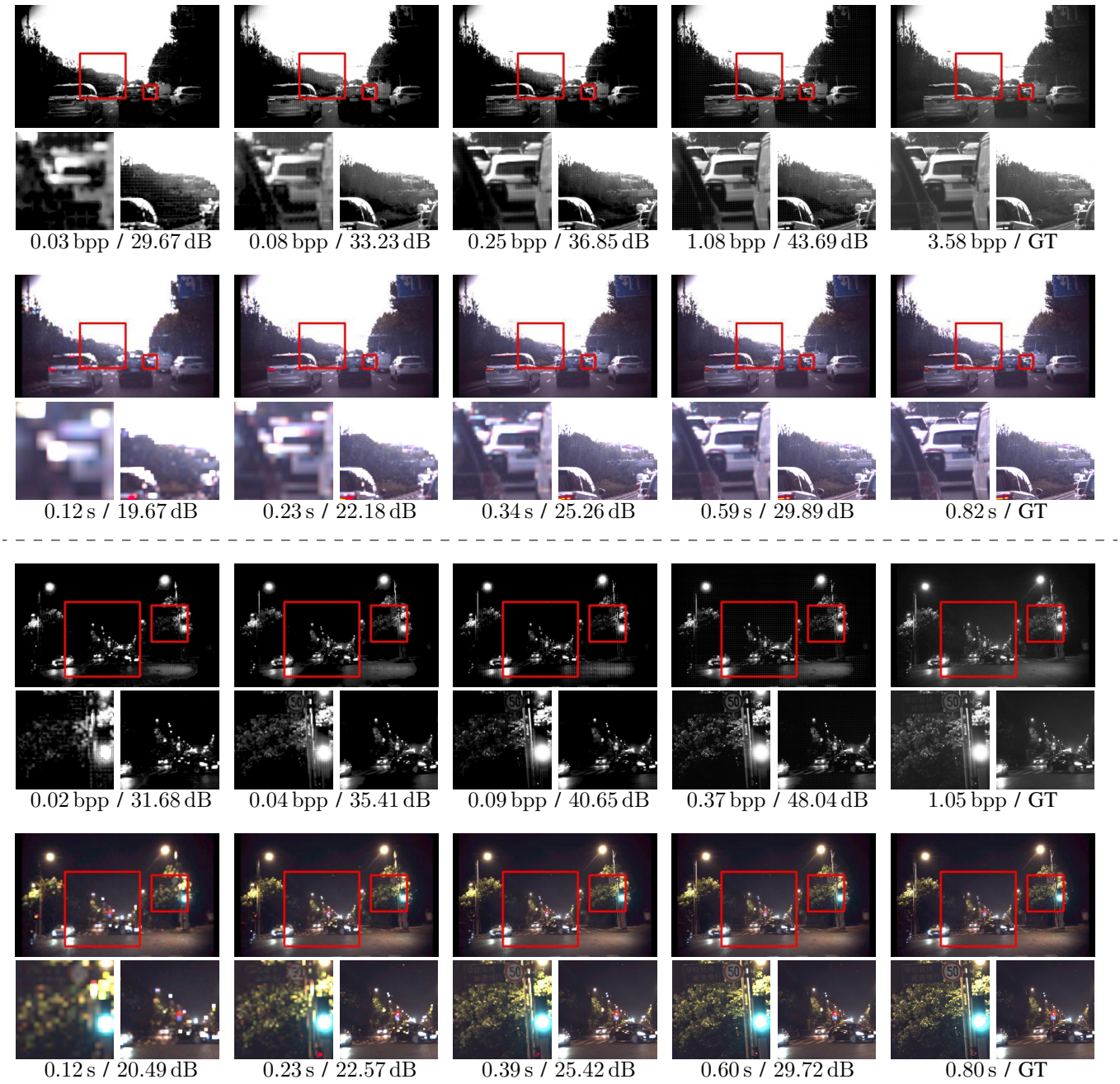


Fig. S11: **Qualitative Visualization of Lossless RIC Progressive Decoding (asi 294mcpro).** The gradual reconstruction of RAW images and their corresponding RGB images converted by an in-camera ISP. Bits per pixel (bpp) / PSNR (dB) is shown under RAW images. Decoding latency (s) / PSNR (dB) is also listed below RGB images. PSNR is derived against the GT (ground truth). Gamma correction and brightness adjustment have been applied for a better view. *Zoom for more details.*