Supplementary Material

5 Motivating Application Scenarios

In Section II,[1] we assumed the position of ROI in the image is known to the transmitter and receiver. In this section, we explain how the ROI position can be obtained as reliably with high accuracy in real scenarios such as autonomous driving and VR/AR. The details are as follows:

[Vehicles] According to a report from the National Highway Traffic Safety Administration (NHTSA), most crashes (more than 100 out of 130 crashes) involving autonomous vehicles are vehicle-to-vehicle collisions. Among these, most of them are secondary collisions, where a following vehicle crashes into a leading vehicle that suddenly brakes due to a sudden obstacle [1, 2].

In such cases, a possible solution proposed in the literature is for the leading vehicle to transmit its front-view scene to the following vehicle in real time, allowing the latter to have sufficient time for decision-making [3]. This paper envisions such a scenario, in which fast tasks such as segmentation and collision probability prediction identify ROI tiles containing the sudden obstacle ahead, which are then transmitted through the ROI-JSCC system to minimize transmission latency.

[VR/AR devices] Most VR/AR headsets are equipped with dedicated sensors or algorithms that track the user's focal point, which corresponds to the ROI patch in our framework [4, 5]. Hence, devices' sensor determines the location of ROI. The hardware provides the position of ROI(focal position), which developers can easily access through the provided SDK to build applications. Thus, we can assume the ROI position is given as the user's focal point, which can be provided with high accuracy by the current commercial devices. This is another motivating scenario of our work to focus only on JSCC-based ROI-image transmission with the known-ROI position assumption.

6 Extensive Experiments

6.1 Main Results

Visual Inspection: Figure 7 compares ROI visualization results for the reconstructed Kodak image at CPP $= \frac{1}{12}$ and SNR = 4dB. In this result, our ROI-JSCC shows the clearest reconstruction with the highest PSNR_{ROI}. For instance, FAJSCC induces color jittering in the text, while SwinJSCC causes blurring near the connection between the wing and the main body. In the case of ConvJSCC, the degradation is so severe that the text becomes barely readable. In contrast, ROI-JSCC effectively reconstructs the image without these artifacts and preserves high ROI quality. In the case of PSNR_{Avg}, compared to the recent two SOTA models, FAJSCC (28.01dB) and SwinJSCC (27.81dB), our ROI-JSCC (27.79dB) shows minimal degradation.

ROI Robustness: Here we discuss the robustness of our ROI-JSCC for various ROI settings. Firstly, without any fine-tuning and retraining, we changed the number of patches to 8×8, which are different from the training setting of 4×4. Figures 8 show performance results for this 8×8

[1] The sections not in this supplementary material indicate the sections of the main paper.

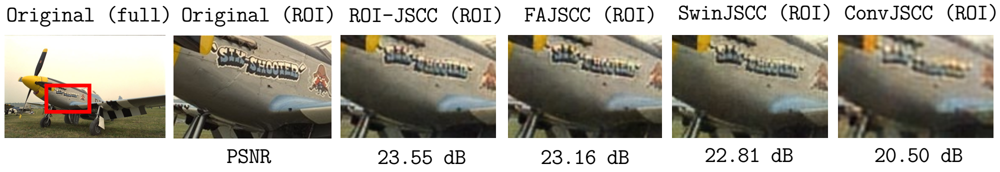


Figure 7: The ROI quality of the reconstructed Kodak image at CPP = 1/12, SNR = 4dB. The red box is the ROI of the transmitted image.

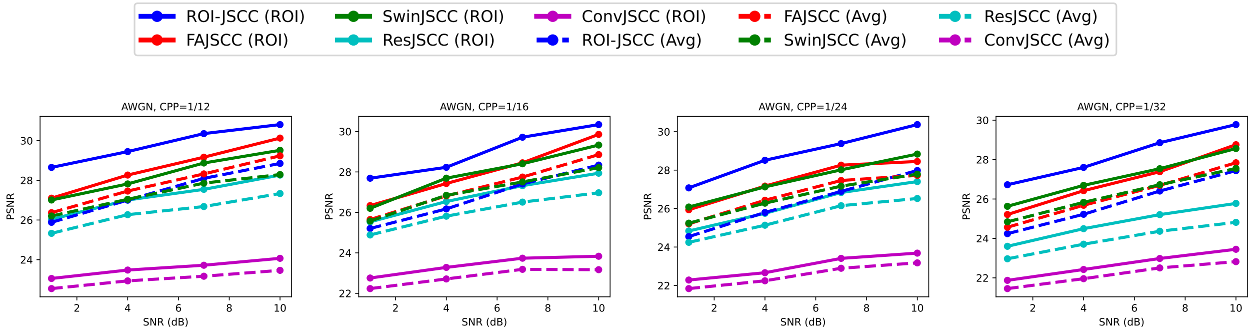


Figure 8: ROI and average PSNR results under different SNR and CPP environments. Different from the main setting, we set $n_h = n_w = 8$, i.e., view image as 8×8 patches.

patches settings. Interestingly, compared to 4×4 patches setting, the ROI performances of our ROI-JSCC show significant improvement while maintaining average performances. This verifies the robustness of our ROI-JSCC for various patch number settings.

This significant performance improvement has two reasons. Firstly, when the number of patches becomes 8×8, the ratio of ROI patch in the full image becomes much smaller. As a result, it is easy to increase ROI performance, since increasing small ROI areas is easier than large ROI areas. Secondly, as the number of patches increases, the probability that a randomly selected patch has only simple patterns becomes higher. When patches are divided into 4×4, the randomly selected patch will have some part of the complex patterns with high probability. Complex patterns, such as animals or statues, are difficult to reconstruct. On the contrary, when patches are divided into 8×8, the probability that a randomly selected patch has complex patterns becomes lower. As a result, more patches will have only simple patterns. Simple patterns, such as sky-like backgrounds, are easy to reconstruct.

In addition to the above discussion for various patch number settings, we discuss various aspects of the ROI Robustness of our ROI-JSCC as follows:

- Variants of ROI Positions: Our experiments are conducted by selecting ROI positions randomly (but same random seed as the other models for fair comparison). Our results show that our ROI-JSCC always shows outstanding ROI performances for different metrics (PSNR and SSIM) under various communication environments (various CPP and SNR settings of AWGN, and fast Rayleigh fading channels). Thus, our ROI-JSCC is robust under varying ROI-positions.
- Resolution Variants of ROI Patches: Note that the width and height sizes of DIV2K, and

Kodak datasets are varying from 1356 to 2040, and from 512 to 768 respectively. Moreover, as the number of patches changes from 4×4 to 8×8, the resolution of ROI patches also becomes different. We verified the performance of our ROI-JSCC in these settings. Thus, our ROI-JSCC is robust under resolution variants of ROI patches.

- Multiple ROIs: Regarding multiple ROIs, we acknowledge that supporting them is necessary for certain applications; however, it represents a nontrivial extension of the current framework. For instance, when considering two ROIs, the model must be trained for all possible ROI pairs, leading to a training complexity proportional to the square of the number of patches. Therefore, directly extending the current ROI-JSCC to handle multiple ROIs is not straightforward. We leave this as a direction for future work.

As a result, our ROI-JSCC is robust for various $n_h \times n_w$ patch settings, ROI positions, and resolution of ROI areas. Although we leave multiple ROI issues as future work, our ROI-JSCC is already able to give useful help for existing applications, especially for VR/AR and collision avoidance of autonomous vehicles that are discussed in Section 5. Note that, in VR/AR settings, we discussed setting humans' foveal areas as ROI areas to increase user experience. By the physiology of the eye, a usual human has only one foveal area [6]. Thus, it is sufficient to only consider single ROI areas in VR/AR applications.

In the case of collision avoidance of autonomous vehicles, supporting only one ROI region may be sufficient. The probability that multiple ROIs are needed to avoid collisions seems low, since it is less likely for several collisions to occur simultaneously. For example, it is difficult to imagine a situation where multiple collision risks arise simultaneously—an object suddenly appears in front of the driver while vehicles in the opposite lanes change toward the driver at the same time.

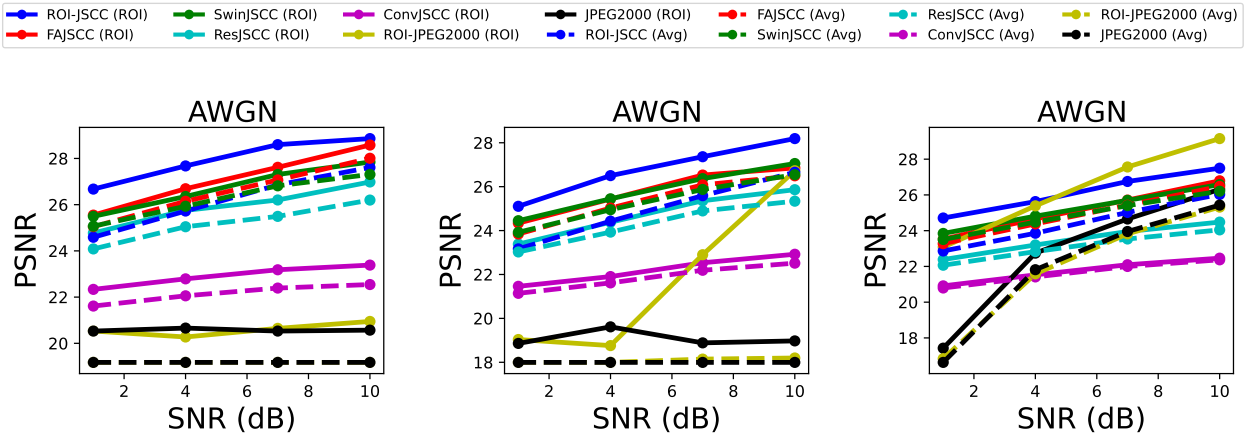
6.2 Comparison with Traditional Approach

In the early 2000s, several studies explored JSCC for regions of interest (ROI). However, these approaches are not true JSCC methods and not suitable for the scenarios considered in our work. Using the classical ROI–JPEG2000–based JSCC as an example, we provide a detailed analysis as follows:

(1) Likewise, other ROI-JSCC of the early 2000s [7,8], the ROI-JPEG2000-based JSCC is not considered a true JSCC in the modern sense. It relies on separate source coding (JPEG2000) and channel coding (e.g., LDPC) schemes, with only weak joint optimization between the JPEG2000 and channel coding (LDPC) parameters.

(2) Based on our implementation of ROI-JPEG2000, its applicability is limited and not aligned with wireless communication and with our target scenarios, such as low-latency scenarios (e.g., autonomous vehicles) and/or computationally efficient devices (e.g., AR/VR).

Figure 9 is our implementation of ROI-JPEG2000 and JPEG2000, assuming the use of optimal channel codes (i.e., hypothetical capacity-achieving codes). JPEG2000 is implemented by the open source library OpenJPEG, and ROI-JPEG2000 by designing an ROI-based bits-usage per



128 × 128 resolution

256 × 256 resolution

512 × 512 resolution

Figure 9: Comparison with deepJSCC methods and JPEG2000s under various image resolutions, and SNRs. The image resolutions are depicted below each figure.

pixel (BPP) allocation mechanism for JPEG2000. The ROI-based BPP allocation mechanism is a modified version of our ROI-JSCC's bandwidth allocation. The detailed implementations are in Appendix A, and B. Then, two observations can be made.

- (ROI-)JPEG2000 codes have a dedicated structure to a specific communication scenario (e.g., SNR, size of images, etc), which means that the encoding and decoding are not universally good. As seen in Figure 9, some (ROI-)JPEG2000 results (black: JPEG2000, yellow: ROI-JPEG2000) exhibit flat behavior over varying SNRs. This occurs when the channel capacity is below the minimum bitstream size required by (ROI-)JPEG2000. This is because of its coding structure, (ROI-)JPEG2000 always requires a certain number of bits to store code's metadata (headers, codeblock information, quantization parameters, packet markers, etc.), even for simple images [9]. Consequently, when the channel capacity falls below this threshold, (ROI-)JPEG2000 encodes the image at its lowest possible rate, exceeding the intended BPP constraint. Therefore, (ROI-)JPEG2000-based JSCC is only applicable when the channel capacity is large enough or the image resolution is high enough for the metadata overhead to remain relatively small. To solve this, the entire code structure must be redesigned for each environment, e.g., SNRs, channels, image sizes, etc, which requires huge extra efforts and hardware complexities.
- Another major drawback of (ROI-)JPEG2000-based JSCC, perhaps even more critical, lies in its computational latency. Since (ROI-)JPEG2000 involves iterative optimization of wavelet parameters, it requires approximately 10 seconds (Xeon(R) CPU E5-2650 v3 used) for encoding and decoding (even without channel coding) of DIV2K high-resolution images on our machine, whereas the proposed ROI-JSCC is much more efficient, taking only about 0.5 seconds (P100 GPU used). Even on the best-performing high-end server computer reported in the literature, JPEG2000 still requires around 60 ms only for encoding [10]. In contrast, on the edge device, our ROI-JSCC is expected to operate in $\sim$ 20 ms for

end-to-end encoding and decoding.[2] Considering hardware performances, the actual latency gap will be much larger when ROI-JPEG and ROI-JSCC are implemented on the same device. Furthermore, because the coding architecture of JPEG2000 is inherently incompatible with parallel processing, its potential for computational acceleration on modern hardware remains limited.

7 Future Research

In this research, we proposed the first ROI-JSCC framework with a detailed model architecture and optimization methods. Moreover, we extensively verified our ROI-JSCC performances under various performance metrics, channels, and ROI settings. Based on our work, appropriate future works are as follows:

- **Multimodal ROI-JSCC for Autonomous vehicles:** In reality, autonomous vehicles aggregate different types of data from cameras, radars, LiDARs. Transmitting these various data with ROI guidance will be much efficient than our current ROI-JSCC. Moreover, in the case of a collision avoidance scenario, our $\log_2(n_h n_w)$ bit length ROI position information also contains some part of the information for collision probability. Finding how to use such an ROI position for the receiver's autonomous driving system is also an interesting research direction.
- **Implementing ROI-JSCC on VR/AR devices:** Current eye tracking technologies provide fast and reliable foveal area [12–14], and using this foveal area as ROI for our ROI-JSCC is sufficient to increase the user experience as we discussed in Section 5. However, integrating eye tracking technologies and our ROI-JSCC can give a synergistic effect to each other. For example, neural features of ROI-JSCC can help estimate the next gaze point trajectory, and the ROI feature processing of our ROI-JSCC can be faster by using values obtained from eye-tracking estimation, rather than processing ROI embedding blocks.

A JPEG2000 with Optimal Channel coding

Like JSCC uses CPP as an available bandwidth resource constraint, source coding uses bits-per-pixel (BPP) as a data resource constraint. By the information theory, the BPP R for reliable data transmission with channel capacity achieving code is as follows [15]:

$$nR \leq kC_{channel},$$

where $n := H \times W \times 3$ is the number of pixels in image $\mathbf{x}$, k is the number of transmitted symbols, and $C_{channel}$ is the channel capacity, which is specified by the channel properties. By dividing n in the both sides, the maximum possible BPP R_{MAX} is as follows:

$$R_{\text{MAX}} = \text{CPP} \times C_{channel}.$$

[2] Inference of YOLOv8s (29.7 GFLOPs) with on-device GPU takes in 7.94 ms [11]. For 2K images, our ROI-JSCC uses $\sim$ 73GFLOPs.

Thus, R_{MAX} can be calculated via the given CPP and $C_{channel}$. Note that $C_{channel} = \log_2(1 + \text{SNR})$ for the complex Gaussian channels that we consider. Since $C_{channel}$ can be achieved via ideal capacity-achieving channel coding and modulation methods, the actually reliably transmitted BPP is lower than R_{MAX}. Thus, our JPEG2000 with R_{MAX} should be one of the upper bounds of traditional coding methods. We implemented JPEG2000 with the open source library OpenJPEG.

B ROI-JPEG2000

Motivated by our ROI-guided bandwidth allocation, we make a new baseline ROI-JPEG2000 to compare with our ROI-JSCC. ROI-JPEG2000 divides image patches based on the ROI map.[3] Then, likewise ROI-based bandwidth allocation proposed in the main paper, we propose ROI-based BPP allocation as follows:

$$\begin{aligned} R_{\text{ROI}} =& (1 + \eta\tau)R_{\text{MAX}}, \\ R_{\text{ROP}} =& R_{\text{MAX}}, \\ R_{\text{RONI}} =& (1 - \eta\tau)R_{\text{MAX}}, \end{aligned}$$

where η and τ are same with the our ROI-JSCC. Likewise ROI-based bandwidth allocation, this ROI-based BPP allocation enhances the ROI quality with only a minimal degradation in average performance.

[3] Remind the ROI map explained in Section II of the main paper

References

[1] N. H. T. S. Administration et al., “Summary report: Standing general order on crash reporting for automated driving systems,” U.S. Dep. Transp., vol. 813, p. 324, Jun. 2022.

[2] C. Wang, F. Chen, Y. Zhang, S. Wang, B. Yu, and J. Cheng, “Temporal stability of factors affecting injury severity in rear-end and non-rear-end crashes: A random parameter approach with heterogeneity in means and variances,” Anal. Methods Accid. Res., vol. 35, p. 100219, Sep. 2022.

[3] R. Zhang, K. Li, Y. Wu, D. Zhao, Z. Lv, F. Li, X. Chen, Z. Qiu, and F. Yu, “A multi-vehicle longitudinal trajectory collision avoidance strategy using aebs with vehicle-infrastructure communication,” IEEE Transactions on Vehicular Technology, vol. 71, no. 2, pp. 1253–1266, Feb. 2022.

[4] S. Wei, D. Bloemers, and A. Rovira, “A preliminary study of the eye tracker in the meta quest pro,” in Proc. ACM Interact. Media Experiences, Jun. 2023, pp. 216–221.

[5] S. Aziz, D. J. Lohr, L. Friedman, and O. Komogortsev, “Evaluation of eye tracking signal quality for virtual reality applications: A case study in the meta quest pro,” in Proc. ACM Symp. Eye Track. Res. Appl., Jun. 2024, pp. 1–8.

[6] L. A. Levin, P. L. Kaufman, and M. E. Hartnett, Adler’s physiology of the eye. Philadelphia, PA, USA: Elsevier Health Sciences, 2024.

[7] Z. Zhang, G. Zhu, F. Wang, and L. Xie, “ROI-based joint source-channel coding for wireless video transmission,” in Proc. Int. Conf. Adv. Commun. Technol. (ICACT), Feb. 2006, pp. 923–926.

[8] Y. Sun, I. Ahmad, D. Li, and Y.-Q. Zhang, “Region-based rate control and bit allocation for wireless video transmission,” IEEE Trans. Multimedia, vol. 8, no. 1, pp. 1–10, Feb. 2006.

[9] D. S. Taubman and M. W. Marcellin, JPEG2000: Image compression fundamentals, standards and practice. New York, NY, USA: Springer Science+Business Media, 2002.

[10] “Benchmarks for jpeg2000 encoders on cpu and gpu,” [Online]. Available: https://www.fastcompression.com/benchmarks/benchmarks-j2k.htm.

[11] “Understanding real-world latency vs. theoretical estimates on jetson orin nx for yolov8s,” Oct. 2024, [Online]. Available: https://myip.kr/nZkdd.

[12] K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba, “Eye tracking for everyone,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2016, pp. 2176–2184.

[13] N. Valliappan, N. Dai, E. Steinberg, J. He, K. Rogers, V. Ramachandran, P. Xu, M. Shojaeizadeh, L. Guo, K. Kohlhoff et al., “Accelerating eye movement research via accurate and affordable smartphone eye tracking,” Nat. Commun., vol. 11, no. 1, p. 4553, Sep. 2020.

[14] N. Chen, Y. Shen, T. Zhang, Y. Yang, and H. Wen, “Ex-gaze: High-frequency and low-latency gaze tracking with hybrid event-frame cameras for on-device extended reality,” IEEE Trans. Vis. Comput. Graph., May 2025.

[15] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, “Deep joint source-channel coding for wireless image transmission,” IEEE Trans. on Cogn. Commun. Netw., vol. 5, no. 3, pp. 567–579, May 2019.