Optimizing JPEG Decoder for Bitstream-Corrupted Image Restoration

Shumin Jiang[1], Hao Qin[1], Tianyi Liu[2], and Yi Wang[1]
[1]Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hong Kong SAR
[2]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
E-mail: shumin.jiang@connect.polyu.hk, yi-eie.wang@polyu.edu.hk

***Abstract*—This paper proposes an image restoration method based on an optimized JPEG decoder, aiming to address the issue of image bitstream corruption. As a fundamental form of image signals, bitstreams are often damaged by various factors (e.g., wireless channel fading, medium aging, and noise interference), resulting in bit flips that severely degrade image quality. Existing image restoration methods mostly focus on the pixel domain, making it difficult to effectively address structural errors caused by bitstream corruption. To better address this challenge, we propose an optimized JPEG decoder that incorporates compressed sensing techniques to restore and reconstruct the image in the image block-level domain. Our approach can more effectively restore the structural damage of an image, rather than being limited to traditional pixel-level restoration. To evaluate the effectiveness of the method, we simulate bit flips on two sets of real images and construct a bitstream-corrupted image dataset that simulates real-world environments. Under different bit error rates (BER), our method (SCA-CS) outperforms the state-of-the-art (SCA) method in repairing image block colors and eliminating block shift problems.**

I. Introduction

Image signals are typically converted into bitstreams through encoding. These bitstreams serve as a compressed representation of the image, carrying key information such as color, texture, and structure. The JPEG [1] compression standard uses lossy algorithms to provide near-psychovisually lossless image quality, significantly reducing data redundancy and improving transmission rates.

However, in practical applications, factors such as wireless channel fading, multipath effects, and storage medium [2], [3] degradation can cause image bitstreams to become corrupted, leading to random bit flips [4], [5]. This, in turn, may result in structural damages such as block misalignment, color distortion, and decoding interruptions. In addition, factors such as motion blur [6], noise interference, and quantization errors can also lead to image degradation. These not only affect the visual quality of the image but also cause serious interference in subsequent analysis and recognition tasks. Existing image restoration methods [7] mainly focus on pixel-level repair of degraded regions in the decoded image. Although these methods are effective, they are challenging to apply to damage in the bitstream caused during encoding or storage transmission. To address this issue, Liu *et al.* [8] proposed a fault-tolerant and robust decoding strategy based on the minimum coding unit (MCU). When decoding fails, this strategy attempts to skip one or more bits in order to realign the boundaries of the entropy-coded symbols, thereby restoring the decoding of subsequent MCUs and significantly repairing structural errors caused by corrupted bitstreams. However, if the resynchronization fails or the skipped position is inaccurate, it may lead to MCU corruption, causing some data to be discarded, which in turn results in information loss and affects the overall image quality and detail restoration.

Existing decoders still lack sufficient robustness when facing large-scale MCU damage. Therefore, we aim to propose a novel decoding framework based on compressed domain restoration, designed to enhance the image's ability to recover from structural damage. Specifically, we restore the global structural information of the image by adjusting the DC frequency domain coefficients. Combined with compressed sensing [9] techniques, we construct an observation matrix and use the Orthogonal Matching Pursuit (OMP) algorithm to perform sparse reconstruction of the AC coefficients, thereby recovering more image details. Experimental results show that this method demonstrates excellent restoration performance and robustness in natural image recovery, significantly outperforming traditional decoders. The main contributions are as follows:

- A hierarchical MCU repair mechanism is designed, employing an adjacent block prediction compensation strategy to restore DC coefficients. The AC coefficients are sparsely reconstructed based on OMP compressed sensing method.
- We have constructed a controllable bit-flip probability model and generated a JPEG-damaged image dataset covering various bit error rates to simulate transmission errors and storage medium degradation issues in real-world scenarios.
- Experiments show that our method achieves better visual quality and objective evaluation metrics under different bit error rate conditions.

II. Related Works

JPEG Decoding: The JPEG standard is a lossy image compression format, with key components including DCT transformation, quantization, and entropy encoding. Decoding [10] is the inverse operation of encoding. The standard JPEG decoder is highly susceptible to bitstream flip errors when processing frequency domain information, leading to decoding failures and severe image damage. Therefore, image restoration becomes a crucial step in ensuring decoding quality, particularly in addressing the structural damage caused by bit errors.

979-8-3315-7206-8/25/$31.00

Bitstream Repair: Numerous studies focus on the repair of corrupted bitstream images. Traditional channel coding [11] achieves error correction through redundancy, but its effectiveness is limited for images lacking redundancy or those that cannot be re-encoded. Packet-level Forward Error Correction [12] improves efficiency in multicast and real-time transmission, but it cannot handle randomly corrupted bitstreams without pre-set redundancy. In the field of digital video, error concealment methods such as temporal interpolation and spatial pixel replication [13], [14] rely on information from adjacent frames or intact blocks. Overall, traditional approaches struggle to handle decoding failures caused by deep corruption due to insufficient redundancy, lack of retransmission, or failure to consider frequency-domain structures. Therefore, there is an urgent need to explore more refined frequency-domain repair strategies to overcome the limitations of existing methods in repairing corrupted images.

Compressed Sensing: Compressed Sensing (CS) is an efficient signal sampling and reconstruction method based on the theory of signal sparsity, widely applied in the field of image coding. The core idea of CS is to design a suitable observation matrix that maps high-dimensional sparse signals to a lower-dimensional observation space, allowing for accurate signal reconstruction from a limited set of observation data. Candes *et al.*, Baraniuk *et al.*, and Ji *et al.* [15]–[17] systematically introduced compressed sensing theory, ushering in a new era in signal processing. This paper uses the OMP [18] algorithm for signal reconstruction. Based on the characteristics of compressed sensing, it reconstructs the AC coefficient components in the damaged MCUs, thereby improving the quality of the decoded image.

III. Method

In this section, we present a robust decoding framework based on compression-domain restoration, referred to as SCA-CS. The baseline SCA (self-compensation and alignment) represents the current state-of-the-art approach. Building upon SCA, our method SCA-CS further incorporates compressed sensing to exploit signal sparsity in the compression domain, thereby achieving improved decoding accuracy and reconstruction quality while maintaining computational efficiency. Fig 1 illustrates the structure of the framework. We focus on repairing the MCUs in entropy encoding, adjusting the DC coefficients and reconstructing the AC coefficients. In the JPEG standard, the image is divided into multiple MCU units for decoding. Since the color space of JPEG images typically uses YCrCb [19], different sampling formats result in varying numbers of 8×8 data units (DU) within the MCU. Each DU undergoes a Discrete Cosine Transform (DCT) transformation, producing a vector of 64 coefficients, with the first coefficient being the DC coefficient, representing the average brightness of the image block. The remaining 63 coefficients are AC coefficients, which represent the frequency information of the block. In subsequent processing, these 64 DCT coefficients will be restored and entropy-decoded.

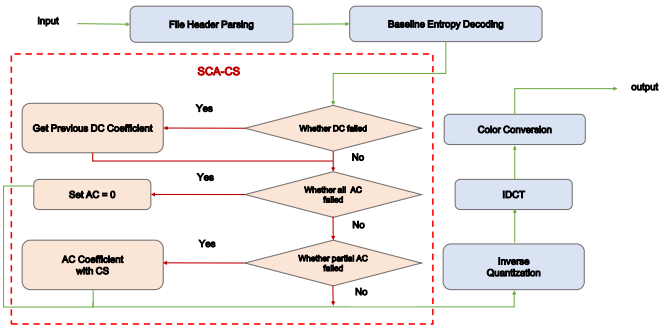

Fig. 1. Overall method flowchart

JPEG decodes each MCU of the image using a lookup table method. If decoding is successful, the current MCU position is calculated, the corresponding coefficients are stored and filled into the image, and decoding continues with the next MCU. If decoding fails, there are two possible scenarios: One scenario is that the decoding has reached a non-data segment or the end of the image, where the current MCU coefficients can be set to zero to avoid generating an abnormal image. Another scenario is that a bitstream flip occurs in the data segment, causing MCU data corruption; in this case, a recovery strategy can be applied to restore the damaged DC and AC coefficients to ensure image integrity.

DC Coefficient Restoration: Since the DC coefficients of adjacent DUs in neighboring MCUs usually exhibit small variations, they often represent similar background or continuous edge information, with smooth transitions in brightness. Therefore, the DC coefficient of the current DU can be accurately predicted and filled using the DC coefficients of the DUs in the neighboring MCUs. In the actual decoding process, when an MCU error occurs, the decoder attempts to read the previous DC coefficient and uses it to fill the current MCU's DC coefficient. It then continues decoding the remaining AC coefficients. This method not only ensures the continuity of the decoding process but also prevents the degradation of image quality caused by the loss of some DC coefficients. JPEG uses differential encoding [20] to store DC coefficients, meaning that the current DC coefficient is not stored as an absolute value but as the difference from the previous block's DC coefficient. Therefore, during the filling process, we need to adjust according to the actual difference. Let the current MCU DC coefficient difference be ΔDC, and the previous DC coefficient be $\mathrm{DC}_{\mathrm{prev}}$, the true coefficient DC can be calculated using the following formula:

$$\mathrm{DC}_{\mathrm{curr}} = \mathrm{DC}_{\mathrm{prev}} + \Delta\mathrm{DC} \tag{1}$$

AC Coefficient Restoration: The core premise of compressed sensing is that the signal exhibits sparsity. For image processing, after the input image undergoes DCT, 64 DCT coefficients can be obtained [21]. Among these, The vast majority of brightness information is concentrated in the low-

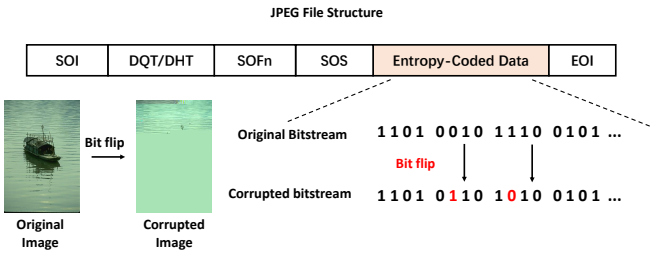

Fig. 2. Bit flips are applied to the entropy-coded data to simulate controllable JPEG bitstream damage

frequency part and takes nonzero values, while the high-frequency AC coefficients are close to or equal to zero. This results in the image exhibiting sparse characteristics in the DCT domain. Therefore, compressed sensing methods can be effectively used to reconstruct the AC coefficients.

During the decoding process, if all AC coefficients of the current MCU are corrupted, no valid AC coefficients can be obtained, and they must be set to zero. If only some of the AC coefficients are damaged and there are decodable valid coefficients, the 64-dimensional DCT coefficients can be treated as the signal vector $\mathbf{x}$ to be restored. The M decoded AC coefficients form the observation vector $\mathbf{y}$, and compressed sensing theory can be applied to reconstruct the signal. In the specific implementation, a 64×64 DCT matrix $\mathbf{D}$ is constructed as the sparse basis $\Psi \in R^{64\times64}$. The observation matrix $\Phi \in R^{M\times64}$ is formed based on the positions of the M decoded AC coefficients and acts as a selection matrix, indicating which coefficients are observed. Based on this, the sparse coefficient vector $\mathbf{s} \in R^{64}$ can be solved using the observation vector $\mathbf{y}$, the sparse basis Ψ, and the observation matrix Φ.

To achieve this, the OMP algorithm is adopted for sparse reconstruction. First, initialize the residual and support set: $\mathbf{r}_0 = \mathbf{Y}$, $\Lambda = \emptyset$. At each iteration, select the index with the highest correlation:

$$j = \arg\max_j \left|\Phi_j^\top \mathbf{r}_{t-1}\right| \tag{2}$$

Add j to the support set Λ, and solve the least-squares problem:

$$\hat{\mathbf{S}}_\Lambda = \arg\min_{\mathbf{S}_\Lambda} \|\mathbf{Y} - \Phi_\Lambda \mathbf{S}_\Lambda\|_2 \tag{3}$$

Iterate to update the residual until it falls below the specified threshold, then stop. The indices in the support set are inserted into a zero vector to obtain the corresponding reconstructed sparse vector $\mathbf{S}$, which is filled into the sparse DCT coefficients to produce the complete signal image $\mathbf{x}$. In this way, not only can the AC coefficients be effectively recovered, but more image details can also be reconstructed.

IV. Experiments

Dataset Construction: We selected two datasets as the foundation for constructing our dataset: BSD500 [22] and our own dataset. Both datasets contain 500 high-quality images encompassing a wide range of scene types, such as natural

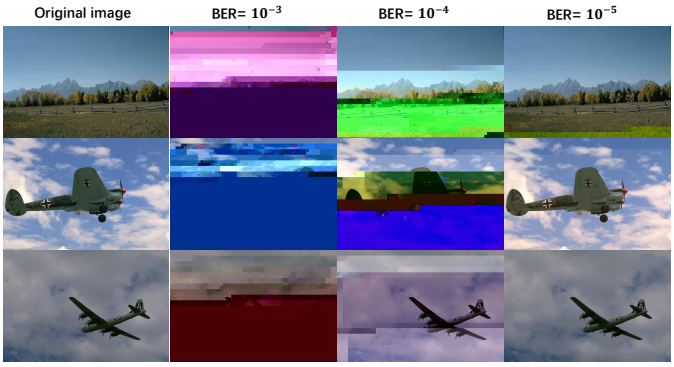

Fig. 3. BSD500 with bitstream corruption under different BER

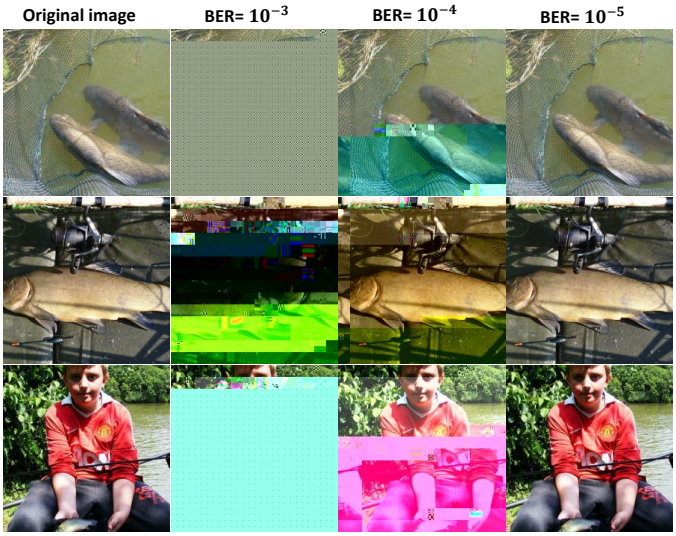

Fig. 4. Our dataset with bitstream corruption under different BER

landscapes, architectural structures, human subjects, and other diverse content. The images in the BSD500 dataset have varying resolutions and sizes, with most being 481×321, and they contain more complex image details and diverse scenes. This dataset can be used to test the performance of the restoration algorithm on more complex images. Our dataset, on the other hand, has a fixed size of 224×224. By testing with this dataset, we can observe the algorithm's performance in restoring textures, details, and local regions of the image. To ensure format consistency, we preprocess the images accordingly before conducting further processing. To simulate the various types of bitstream damage that JPEG images may experience in real-world environments, we introduced a bitstream flip mechanism as shown in Figs. 2. This mechanism controls the severity of the damage to the

TABLE I
Performance of SCA-baseline on BSD500 and Our Dataset

Dataset	BER=10^{-3}	BER=10^{-4}	BER=10^{-5}
BSD500	9.88 / 0.2050	11.23 / 0.2767	13.69 / 0.5380
Our dataset	10.92 / 0.3422	13.53 / 0.6505	15.30 / 0.8249

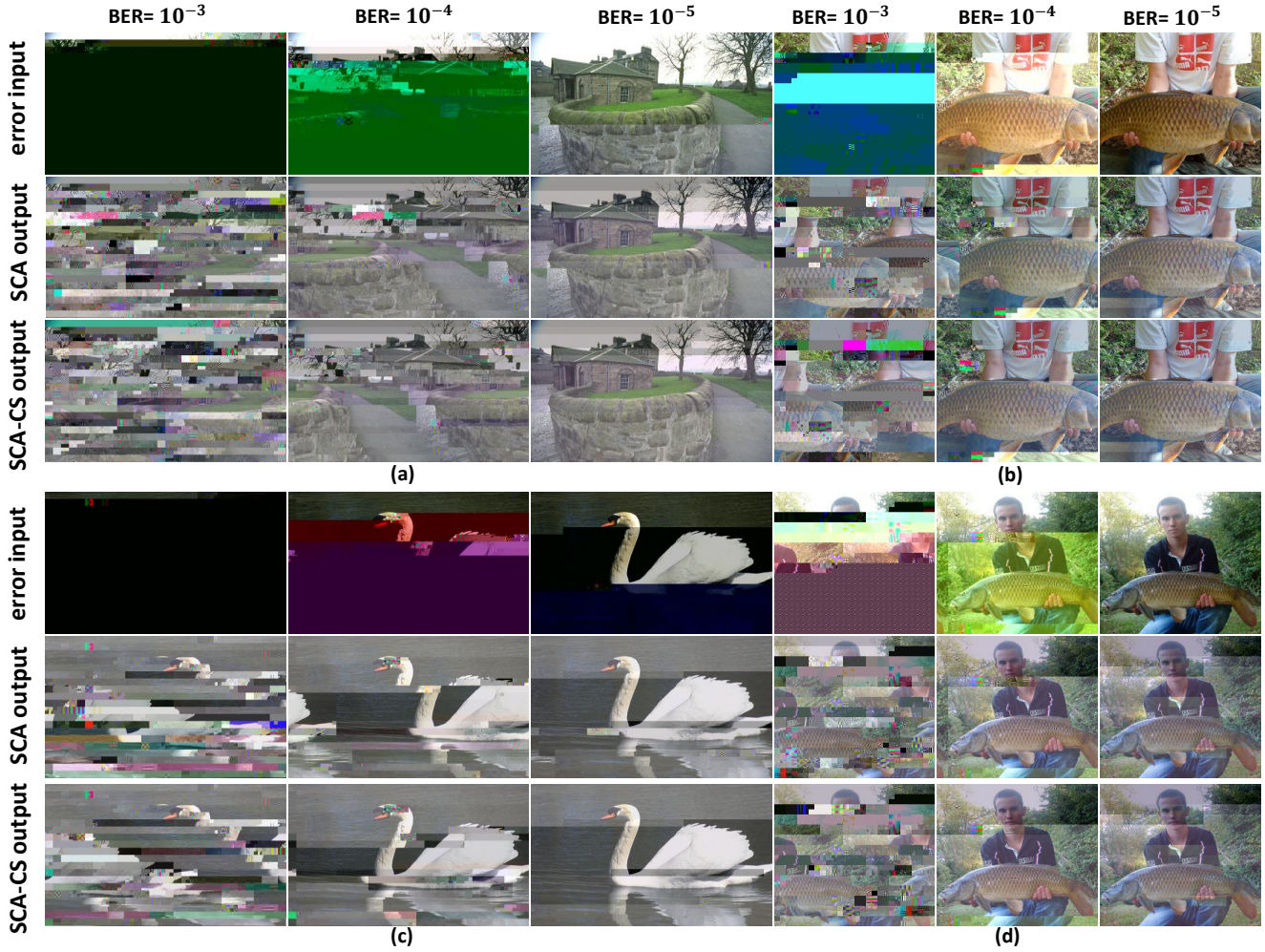

Fig. 5. Comparison of the BSD500 and our dataset after bitstream corruption. SCA refers to Liu's method [8], while SCA-CS represents our method. (a)(c): BSD dataset, (b)(d): our dataset

TABLE II
COMPARISON OF PSNR AND SSIM ON BSD500 DATASET AND OUR DATASET

Dataset	Methods	PSNR / SSIM in different BER		
		10^{-3}	10^{-4}	10^{-5}
BSD500	SCA [8]	10.43 / 0.2041	11.37 / 0.2811	13.80 / 0.5480
	SCA-CS	**10.52 / 0.2070**	**11.47 / 0.2928**	**14.05 / 0.5790**
Our dataset	SCA [8]	10.98 / 0.3455	13.61 / 0.6610	15.31 / 0.8263
	SCA-CS	**11.03 / 0.3515**	**13.66 / 0.6669**	**15.31 / 0.8274**

SCA: Liu's method [8], SCA-CS: our method. The unit of PSNR is dB.

image by adjusting the flip probability parameters, covering scenarios ranging from slight distortion to severe damage. BER (Bitstream Error Rate) is an important metric for measuring the error rate in a data transmission system. It represents the proportion of erroneous bitstreams to the total number of bitstreams transmitted within a unit of time. Assuming a BER of 10^{-5}, it means that approximately 1 bitstream error occurs for every 100,000 bits transmitted by the system.

$$\text{BER} = \frac{\text{Number of Erroneous Bits}}{\text{Total Number of Transmitted Bits}} \tag{4}$$

We introduces PSNR and SSIM as image quality metrics [23] to objectively and quantitatively reflect image quality. As shown in Figs. 3 and 4, since bitstream errors are randomly distributed, they may cause varying degrees of image damage. If the error happens to occur in an unimportant area (such as a flat background with few details), then even with a large BER value, the damage to the image may not be obvious. As a result, images with higher BER may sometimes appear less damaged than those with lower BER.

Zeroing Baseline Analysis: To prevent misdecoding and structural misalignment in SCA under high BER conditions,

TABLE III
COMPLEXITY OF DIFFERENT DECODERS

Dataset	Methods	Running Time (s)	Memory Usage (KB)	Peak Memory Usage (KB)
BSD500	SCA [8]	522.92	870.31	934.88
	SCA-CS	**429.15**	**506.52**	**571.18**
Our dataset	SCA [8]	68.47	878.80	956.52
	SCA-CS	**61.99**	**507.12**	**585.13**

SCA: Liu's method [8], SCA-CS: our method of MCU recovery. The running time is measured in seconds for processing 500 images, while the units for memory usage and peak memory usage are in KB.

this paper introduces a conservative MCU Zeroing Strategy (SCA-baseline). When an MCU decoding failure occurs, this method sets all pixels in the block to zero, preventing the spread of errors and ensuring stability in the recovery process. Table I presents the PSNR and SSIM results on BSD500 and our dataset, demonstrating the stability and simplicity of this approach. As a fault-tolerant baseline, it provides a reference for performance improvement in future methods. While the SCA-baseline retains some structural integrity under low BER, it discards significant image information, leading to noticeable voids. To address this, the paper proposes SCA-CS, which leverages valid bitstream data to recover more details without error propagation.

MCU Recovery: In terms of visual results, we compared the reconstruction performance of the SCA method and the proposed SCA-CS method. It can be observed that under medium-to-low bit error rates, the conventional SCA method often exhibits two prominent issues: (i) some corrupted blocks cannot be effectively repaired, leading to obvious missing regions; and (ii) the overall image tends to suffer from misalignment or shifts, which undermine structural consistency [24] . These problems are particularly pronounced in scenes with complex textures and rich details. In contrast, our method not only enables more corrupted blocks to be effectively repaired but also recovers textures and structural contours more faithfully, significantly reducing block misalignment artifacts. For example, as shown in Figs. 5, the results reconstructed by SCA-CS exhibit superior edge continuity, detail preservation, and background smoothness compared with SCA. Therefore, the visual results not only provide intuitive evidence of the advantages of SCA-CS but also further validate the effectiveness and necessity of joint DC and AC coefficient-based restoration.

We evaluated the performance of SCA and SCA-CS in terms of image quality (PSNR and SSIM), as shown in Tables II. For the PSNR metric, the PSNR values of our dataset either improved or remained unchanged. On the BSD500 dataset, as the BER decreases, the PSNR increased from 0.4% to 1%, demonstrating that our restoration strategy effectively suppresses error propagation under different conditions, showcasing its performance advantages. For SSIM, both datasets experienced a slight decrease under extreme transmission conditions, likely due to local structural disturbances introduced by the repair of high-frequency details. However, as the BER decreases, the SSIM of the BSD500 dataset gradually improved by up to 5.6%, and our dataset also saw an average improvement of 1%, significantly enhancing the image reconstruction quality.

At low bit error rates, the color degradation in some areas of the image can be attributed to the varying performance of the restoration algorithm across different regions. While SCA-CS effectively recovers damaged AC coefficients, especially in regions with simpler structures and textures, overcompensation or erroneous recovery may occur in areas with more complex color information or finer details, leading to local color distortion. Specifically, filling DC coefficients may not adequately restore chroma information, causing color degradation in certain regions. This highlights the need for further optimization of the restoration algorithm to avoid local distortions and improve the overall stability and accuracy of color recovery.

The structural integrity and completeness of the image were significantly improved through MCU recovery. Particularly under more severe image damage, our method effectively restores image details. It demonstrates higher robustness under high bit error rate conditions, enhancing both the usability and decoding quality of the image.

We also compared the performance of the two decoders in terms of runtime and memory usage as shown in Tables III. The results show that, on both datasets, our method reduced the runtime compared to SCA, improving computational efficiency. At the same time, it significantly reduced memory usage and peak memory consumption, effectively minimizing resource usage. These results indicate that our method not only enhances image quality but also provides higher efficiency, making it a highly practical and efficient solution.

V. CONCLUSION

We proposed a novel decoder framework, SCA-CS, which effectively restores images damaged by bitstream errors. We created two datasets and performed random bit flips at different error rates on the bitstream level to simulate the interference encountered during network transmission in real-world scenarios. Additionally, we extended the input image range from color images to grayscale images, enhancing the practicality of the decoder. To improve the quality of images with structural errors, we performed MCU recovery. When a DC error occurred, the previous correct DC coefficient was used to fill in the current MCU, thereby recovering more image information. We also introduced a compressed sensing algorithm to reconstruct the damaged AC coefficients, effectively restoring the damaged values and significantly enhancing the image reconstruction quality. Experimental results verified that our SCA-CS achieved advanced and stable performance in handling structurally damaged JPEG images.

REFERENCES

[1] S. N. Kumar, M. V. Bharadwaj, and S. Subbarayappa, "Performance comparison of jpeg, jpeg xt, jpeg ls, jpeg 2000, jpeg xr, hevc, evc and vvc for images," in *2021 6th International Conference for Convergence in Technology (I2CT)*, IEEE, 2021, pp. 1–8.

[2] D. B. Kurka and D. Gündüz, "Bandwidth-agile image transmission with deep joint source-channel coding," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 8081–8095, 2021.

[3] S. Ghandeharizadeh, S. Irani, and J. Lam, "On configuring a hierarchy of storage media in the age of nvm," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, IEEE, 2018, pp. 1380–1383.

[4] H. Qin, H. Sun, and Y. Wang, "A byte-based gpt-2 model for bit-flip jpeg bitstream restoration," in *2024 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, IEEE, 2024, pp. 1–6.

[5] T. Liu, K. Wu, Y. Wang, W. Liu, K.-H. Yap, and L.-P. Chau, "Bitstream-corrupted video recovery: A novel benchmark dataset and method," *Advances in Neural Information Processing Systems*, vol. 36, pp. 68 420–68 433, 2023.

[6] S. Mondal, S. Das, and P. Ghosh, "Non-blind and blind deconvolution methodologies in restoration of motion-blurred images," in *2024 International Conference on Big Data Analytics in Bioinformatics (DABCon)*, IEEE, 2024, pp. 01–06.

[7] J. Varghese, S. Subash, N. Tairan, and B. Babu, "Laplacian-based frequency domain filter for the restoration of digital images corrupted by periodic noise," *Canadian journal of electrical and computer engineering*, vol. 39, no. 2, pp. 82–91, 2016.

[8] W. Liu, Y. Wang, K.-H. Yap, and L.-P. Chau, "Bitstream-corrupted jpeg images are restorable: Two-stage compensation and alignment framework for image restoration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9979–9988.

[9] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[10] S. Huo and J. Liu, "A high performance jpeg decoding algorithm based on opencl," in *2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, IEEE, 2022, pp. 711–714.

[11] T. Venugopal and S. Radhika, "A survey on channel coding in wireless networks," in *2020 International Conference on Communication and Signal Processing (ICCSP)*, 2020, pp. 0784–0789. DOI: 10.1109/ICCSP48568.2020.9182213.

[12] L. Liu and X. Dong, "Evaluating packet-level forward error correction: 1-d interleaved parity codes," in *2012 8th International Conference on Computing Technology and Information Management (NCM and ICNIT)*, vol. 1, 2012, pp. 370–375.

[13] G. A. Z, A. Jain, and V. K. Sharma, "Image error concealment method by hiding a copy of the same image in it in spatial and wavelet domain before transmission," in *2022 International Conference on Automation, Computing and Renewable Systems (ICACRS)*, 2022, pp. 851–853. DOI: 10.1109/ICACRS55517.2022.10029016.

[14] S. Aign and K. Fazel, "Temporal and spatial error concealment techniques for hierarchical mpeg-2 video codec," in *Proceedings IEEE International Conference on Communications ICC '95*, vol. 3, 1995, 1778–1783 vol.3. DOI: 10.1109/ICC.1995.524505.

[15] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[16] R. G. Baraniuk, "Compressive sensing [lecture notes]," *IEEE signal processing magazine*, vol. 24, no. 4, pp. 118–121, 2007.

[17] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on signal processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[18] J. Tan, "Image processing technology based on omp reconstruction optimization algorithm," *Journal of Computational Methods in Science and Engineering*, vol. 24, no. 3, pp. 1741–1753, 2024.

[19] J. Zou, R. Song, and T. Chen, "A novel image segmentation algorithm based on the microarray camera and modified ycrcb color space," in *2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, IEEE, 2018, pp. 146–149.

[20] M. Dimopoulou, E. G. San Antonio, and M. Antonini, "A jpeg-based image coding solution for data storage on dna," in *2021 29th European Signal Processing Conference (EUSIPCO)*, IEEE, 2021, pp. 786–790.

[21] S. K. Ahmed and S. N. Al-Faydi, "A novel invisible image watermarking based on the relation between the selected dct coefficients," in *2024 2nd International Conference on Software Engineering and Information Technology (ICoSEIT)*, IEEE, 2024, pp. 108–113.

[22] D. Radovic, V. Maksimovic, and B. Jaksic, "Comparative analysis of standard image segmentation methods," in *2025 24th International Symposium INFOTEH-JAHORINA (INFOTEH)*, IEEE, 2025, pp. 1–5.

[23] Y. Zheng and Z. Qin, "Objective image fusion quality evaluation using structural similarity," *Tsinghua Science & Technology*, vol. 14, no. 6, pp. 703–709, 2009.

[24] T. Liu, K. Wu, C. Cai, Y. Wang, K.-H. Yap, and L.-P. Chau, *Towards blind bitstream-corrupted video recovery via a visual foundation model-driven framework*, 2025. arXiv: 2507.22481 [eess.IV]. [Online]. Available: https://arxiv.org/abs/2507.22481.