ANZIAM J. **68, e1** (2026), 1–19
doi:10.1017/S1446181125100242

A PARALLELIZABLE LOSSLESS IMAGE COMPRESSION ALGORITHM FOR STANDARD AND PATHOLOGY IMAGES

HONGFANG YUAN[1], SONGHAI DENG[1], XIANGKAI LIAN✉[2] and ZHENG'AN YAO[1]

(Received 15 December, 2024; accepted 15 October, 2025)

Abstract

This paper introduces a parallelizable lossless image compression algorithm designed for three-channel standard images and two-channel pathology images. The proposed algorithm builds on the Quite OK Image Format (QOI) by addressing its limitations in parallelizability and compression efficiency, thereby enhancing both the compression ratio and processing speed. By incorporating image context and optimizing pixel traversal sequences, the algorithm enables effective parallel processing, achieving rapid compression of million-pixel pathology images within milliseconds, and is scalable to larger whole-slide images. It also delivers exceptional performance in terms of both speed and compression ratio for standard images. Additionally, the low complexity lossless compression for images (LOCO-I) context prediction algorithm used in joint photographic experts group lossless standard (JPEG-LS) is parallelized to improve compression efficiency and speed. By implementing full-process parallelization across the entire compression workflow rather than confining parallelization to individual steps, this approach significantly enhances overall time performance.

2020 *Mathematics subject classification*: primary 68U10; secondary 94A08.

Keywords and phrases: lossless image compression, parallelization.

1. Introduction

Lossless image compression remains critical in imaging pipelines where exact pixel fidelity is mandatory—such as in scientific visualization, medical imaging and digital archiving. Classical schemes span several paradigms: entropy coding [5, 12]; predictive coding exemplified by the joint photographic experts group (JPEG), JPEG lossless standard (JPEG-LS) and portable network graphics (PNG) [1, 19, 20]; transform coding as in JPEG 2000 [15, 18]; and context-adaptive methods such as context-based adaptive lossless image coding (CALIC) [21, 22] (see also general surveys [10]).

[1]School of Mathematics, Sun Yat-sen University, Guangzhou, P. R. China;
e-mail: yuanhf7@mail2.sysu.edu.cn, dengsh7@mail2.sysu.edu.cn, mcsyao@mail.sysu.edu.cn
[2]College of Intelligence Science and Technology, National University of Defense Technology, Changsha, P. R. China; e-mail: lianxiangkai@nudt.edu.cn
© The Author(s), 2026. Published by Cambridge University Press on behalf of The Australian Mathematical Publishing Association Inc.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

The Quite OK Image Format (QOI) [17] is a recent, minimalist and fast lossless codec that achieves compression ratios comparable to PNG while being typically 3 to 4 times faster in both encoding and decoding. Despite its efficiency, certain design choices—particularly, sequential pixel processing and the index mode—can limit performance on high-resolution or pathology images, motivating the development of improved variants.

1.1. Overview of the QOI algorithm To motivate our proposed variants, we briefly review the standard QOI encoding mechanisms. The algorithm compresses 8-bit per channel images—red, green, blue (RGB) or red, green, blue, alpha (RGBA)—by processing pixels sequentially and selecting one of six operational modes based on the current pixel and its context.

TABLE 1. QOI_OP_RUN encoding.

Byte[0]
7 6 5 4 3 2 1 0
1 1 **run**

1.1.1. Run-length encoding Consecutive identical pixels are encoded in run-length mode, using a 6-bit counter combined with the 2-bit flag. This is the most efficient compression mode in QOI. (See Table 1.)

TABLE 2. QOI_OP_INDEX encoding.

Byte[0]
7 6 5 4 3 2 1 0
0 0 **index**

1.1.2. Hash encoding A 64-entry hash table stores recently encountered unique pixels. The hash is computed as

$$hash = (r \cdot 3 + g \cdot 5 + b \cdot 7 + a \cdot 11) \bmod 64.$$

If the current pixel matches the table entry, the "00" flag and 6-bit index are emitted; otherwise, the table is updated. (See Table 2.)

TABLE 3. QOI_OP_DIFF encoding.

Byte[0]
7 6 5 4 3 2 1 0
0 1 **drdgdb**

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

1.1.3. Pixel differential encoding If the alpha channel is unchanged and RGB differences (dr, dg, db) with the previous pixel are in $[-2, 1]$, each difference is encoded with 2 bits and combined with the "01" flag, achieving a compression ratio of 1/4. (See Table 3.)

TABLE 4. QOI_OP_LUMA encoding.

Byte[0]	Byte[1]
7 6 5 4 3 2 1 0	**7 6 5 4 3 2 1 0**
1 0 **diff green**	**dr - dgdb - dg**

1.1.4. Chroma differential encoding This mode exploits the green channel as a luminance reference. If $\Delta G \in [-32, 31]$ and $\Delta R - \Delta G, \Delta B - \Delta G \in [-8, 7]$, values are encoded into two bytes with the "10" flag, achieving a compression ratio of 1/2. (See Table 4.)

TABLE 5. QOI_OP_RGB encoding.

Byte[0]	Byte[1]	Byte[2]	Byte[3]
1 1 1 1 1 1 1 0	**red**	**green**	**blue**

TABLE 6. QOI_OP_RGBA encoding.

Byte[0]	Byte[1]	Byte[2]	Byte[3]	Byte[4]
1 1 1 1 1 1 1 1	**red**	**green**	**blue**	**alpha**

1.1.5. Direct encoding Direct encoding stores the original channel values completely. Flags `0xFE` and `0xFF` distinguish RGB-only or RGBA storage. (See Tables 5 and 6.)

1.2. Limitations of QOI While elegant and fast, QOI has two main limitations for modern high-resolution workloads: (i) limited pixel-level parallelism due to sequential dependencies; and (ii) suboptimal compression for natural and pathology images where the index mode is a bottleneck. Parallel CPU/GPU implementations and GPU-oriented codecs [2, 3, 14, 16] partially alleviate throughput issues but increase architectural complexity. Learning-based approaches [8, 11, 13] improve modelling but are resource-intensive for ultra-high-resolution images (10^8–10^{10} pixels).

1.3. Recent works Recent advances address lossless image coding from modelling and system perspectives. Fitted neural lossless image compression (FNLIC) fits channel-wise discrete mixture models with lightweight heads [23],

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

content-adaptive learning for lossless image compression (CALLIC) adapts entropy models to image content [6] and large language model (LLM)-driven entropy modelling achieves state-of-the-art results while maintaining domain flexibility [4]. Foundation-model-scale approaches such as lossless data compression by large models (LMCompress) explore large-model-based general-purpose compression [7]. For whole-slide imaging, the whole-slide-image lossless compression framework (WISE) [9] introduces tile-wise parallelism, and segmentation-assisted multi-entropy models for learned lossless image compression (SEEC) [24] incorporate semantic masks—a concept complementary to our proposed DIFF/DIFF UP contexts.

1.4. Contributions Building upon QOI [17] and the LOCO-I predictor underlying JPEG-LS [20], we propose two QOI-compatible and fully parallelizable variants: QOI3 for 8-bit RGB images and QOI2 for 8-bit two-channel pathology images. Composite pipelines—QOI3JPEGLS and QOI2JPEGLS—prepend LOCO-I prediction to our QOI variants, eliminating the index mode, introducing a DIFF UP mode to exploit vertical context and optimizing traversal order for full parallelism. Empirically, these methods improve compression ratios on natural and pathology images while preserving QOI-style simplicity and speed.

The contributions are summarized as follows.

(i) Full-process parallelization: algorithmic modifications that remove serial bottlenecks and scale efficiently across threads and cores.
(ii) Specialized variants: QOI3 for RGB and QOI2 for two-channel pathology images, exploiting inter-channel statistical structure.
(iii) Composite pipelines: integration of LOCO-I with QOI variants to reduce residual entropy, maintaining strict losslessness and reversibility.

2. Proposed algorithms

2.1. Lossless compression algorithm for 8-bit three-channel images (QOI3)

This algorithm takes an 8-bit three-channel image as input and encodes pixels in a sequential order: the first pixel, the first column, the first row, and then the remaining pixels from left to right and top to bottom. The encoding uses five modes: **RUN MODE**, **DIFF MODE**, **DIFF UP MODE**, **LUMA MODE** and **RGB MODE**. These modes are organized in ascending order of byte usage and exploit differences in the three channels of neighbouring pixels to optimize compression under both best- and worst-case scenarios.

QOI3 is built upon the original QOI algorithm with several enhancements. It captures contextual relationships in 8-bit three-channel images more effectively by replacing the original INDEX MODE with a newly designed DIFF UP MODE and by optimizing pixel traversal to enable parallel processing. While INDEX MODE in standard QOI performs well for images with many discontinuously repeated pixels (for example, screenshots of phone memos), it often becomes the limiting factor for compression efficiency on natural and pathology images.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

2.1.1. RUN MODE **RUN MODE** implements run-length encoding, which compresses consecutive identical pixels. The maximum run-length is $run = 62$ because $0b11111110$ is reserved as the RGB MODE tag. (See Algorithm 1.)

Algorithm 1 RUN MODE.

Input: Pixel sequence of the 8-bit three-channel image
Initialization: Previous pixel value *prev_px*, run-length counter $run_length = 0$
for each pixel *px* in the image **do**
    if $px == prev_px$ **then**
        $run_length \mathrel{+}= 1$
        if $run_length == 62$ **then** ▷ Maximum run-length due to encoding limitations
            Output the current run-length encoding and reset $run_length = 0$
        end if
    else
        if $run_length > 0$ **then**
            Output the current run-length encoding
        end if
        Update $prev_px = px$
        $run_length = 1$ ▷ Start counting from the new pixel
    end if
end for
if $run_length > 0$ **then**
    Output the remaining run-length encoding
end if
Output: Encoded data of 1 byte length, $\mathbf{b11}||run_length - 1$

2.1.2. DIFF MODE **DIFF MODE** encodes the difference between the current pixel and the pixel to its left. It leverages the fact that differences among the three channels are often small. (See Algorithm 2.)

Algorithm 2 DIFF MODE.

Input: Pixel sequence of the 8-bit three-channel image
Compute differences: $dR = curr_px.R - prev_px.R$, $dG = curr_px.G - prev_px.G$, $dB = curr_px.B - prev_px.B$
if $-2 \le dR, dG, dB \le 1$ **then**
    $dR = dR + 2, dG = dG + 2, dB = dB + 2$ ▷ Offset differences to be non-negative
    Generate encoded output: $\mathbf{b01}||dR||dG||dB$
    Update $prev_px = curr_px$
end if
Output: Encoded difference data of 1 byte length

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

2.1.3. DIFF UP MODE **DIFF UP MODE** encodes the difference between the current pixel and the pixel above it. This mode is similar in principle to DIFF MODE and is used when DIFF MODE is inapplicable, aiming to encode a pixel within a single byte. (See Algorithm 3.)

Algorithm 3 DIFF UP MODE.

Input: Pixel sequence of the 8-bit three-channel image
Compute differences: $dR1 = curr_px.R - up_px.R$, $dG1 = curr_px.G - up_px.G$, $dB1 = curr_px.B - up_px.B$
if $-2 \le dR1, dG1, dB1 \le 1$ **then**
    $dR1 = dR1 + 2$, $dG1 = dG1 + 2$, $dB1 = dB1 + 2$
    Generate encoded output: $\mathbf{b00}||dR1||dG1||dB1$
    Update $prev_px = curr_px$
end if
Output: Encoded difference data of 1 byte length

2.1.4. LUMA MODE **LUMA MODE** encodes pixels whose difference from the previous pixel is large. It exploits the property that channel differences relative to the green channel are often small, allowing compression of larger differences using only two bytes. In most cases, LUMA MODE is rarely activated; for the tested images, fewer than 5% of pixels required this mode. (See Algorithm 4.)

Algorithm 4 LUMA MODE.

Input: Pixel sequence of the 8-bit three-channel image
Compute green difference: $dG = curr_px.G - prev_px.G$
Compute red and blue differences relative to green: $dR_dG = (curr_px.R - prev_px.R) - dG$, $dB_dG = (curr_px.B - prev_px.B) - dG$
if $-32 \le dG \le 31$ **and** $-8 \le dR_dG, dB_dG \le 7$ **then**
    $dG = dG + 32$, $dR_dG = dR_dG + 8$, $dB_dG = dB_dG + 8$ ▷ Offset differences to be nonnegative
    Generate encoded output: $\mathbf{b10}||dG||dR_dG||dB_dG$
    Update $prev_px = curr_px$
end if
Output: Encoded luma difference data of 2 bytes length

2.1.5. RGB MODE **RGB MODE** encodes pixels when the difference from the previous pixel is too large for DIFF or LUMA MODE. It is the least frequently used mode for most natural images. (See Algorithm 5.)

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

Algorithm 5 RGB MODE.

Input: Pixel sequence of the 8-bit three-channel image
Encode as **b11111110**$||R||G||B$
Output: RGB pixel encoding of 4 bytes length

2.1.6. Lossless compression algorithm for 8-bit three-channel images This encoding combines the five compression modes mentioned earlier in sequence to achieve a correct encoding algorithm. (See Algorithm 6.)

Algorithm 6 Lossless compression for 8-bit three-channel images.

Input: 8-bit three-channel image (indexed from 0)
Obtain the number of pixels in length L and width W; denote the encoding as BM, which is an array of integers each ranging from 0 to 255
Initialize a 12-byte header $BM = header = [ASCII(QOI3), 4byte(L), 4byte(W)]$
Add the first pixel encoded in **RGB MODE** to BM: [**b**11111110, R, G, B]
for $l = 1, w = 0$ from left to right until $l = L - 1, w = 0$ **do**
    Extend BM using QOI
end for
for $l = 0, w = 1$ from top to bottom until $l = 0, w = W - 1$ **do**
    From left to right, treating as top to bottom, extend BM using QOI
end for
for $l = 1, w = 1$ from left to right, from top to bottom until $l = L - 1, w = W - 1$ **do**
    Denote $dR = R[l, w] - R[l - 1, w], dG = G[l, w] - G[l - 1, w]$, $dB = B[l, w] - B[l - 1, w]$
    if $dR = dG = dB = 0$ **then**
        Extend BM using **RUN MODE**
    else if $-2 \leq dR, dG, dB \leq 1$ **then**
        Extend BM using **DIFF MODE**
    else if $-2 \leq R[l, w] - R[l, w - 1], G[l, w] - G[l, w - 1], B[l, w] - B[l, w - 1] \leq 1$ **then**
        Extend BM using **DIFF UP MODE**
    else if $-32 \leq dG \leq 31, -8 \leq dR - dG, dB - dG \leq 7$ **then**
        Extend BM using **LUMA MODE**
    else
        Extend BM using **RGB MODE**
    end if
end for
Output: Encoded array BM

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

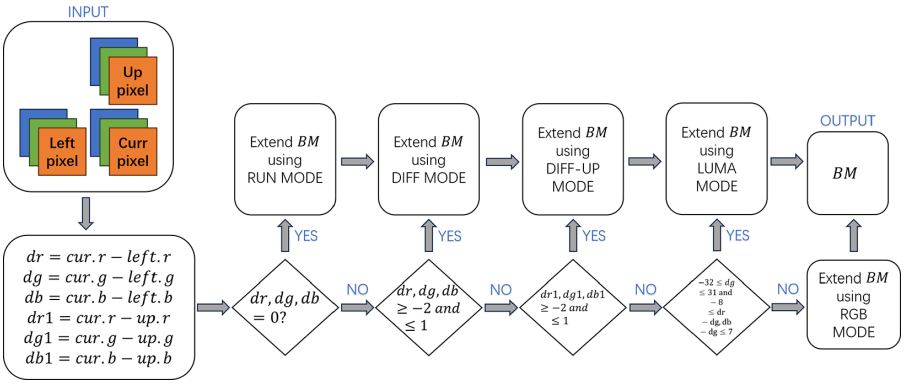


FIGURE 1. QOI3 lossless image compression algorithm.

The functionality of the QOI3 algorithm is visually summarized in Figure 1.

2.2. Lossless compression algorithm for 8-bit two-channel pathology images (QOI2) This algorithm is inspired by our proposed 8-bit two-channel lossless compression design and the original QOI algorithm. It integrates the key principles of both into the compression of 8-bit two-channel images, and hence named QOI2. Most pathology images are stained with two dyes, generating two distinct channels, which motivates the development of a dedicated two-channel compression method. QOI2 is particularly suitable for efficient compression of such pathology images.

2.2.1. RUN MODE **RUN MODE** implements run-length encoding, compressing consecutive identical pixels. The maximum run-length is $run = 63$, as $0b11111111$ is reserved for the RG MODE tag. (See Algorithm 7.)

Algorithm 7 RUN MODE.

1: **Input:** Pixel sequence to be encoded
2: Encode the number of consecutive repeated pixels, up to a maximum of 63. If the run exceeds this value, reset $run = 0$ and continue encoding
3: Encode as $\mathbf{b11}||run - 1$, where $1 \leq run \leq 63$
4: **Output:** Encoded data of 1 byte length

2.2.2. DIFF MODE **DIFF MODE** encodes the difference between the current pixel and its left neighbour, leveraging the fact that differences across channels are typically small. (See Algorithm 8.)

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

Algorithm 8 DIFF MODE.

1: **Input:** Pixel sequence to be encoded
2: Compute differences from the left pixel: $dR = px_R - left_px_R$, $dG = px_G - left_px_G$
3: Adjust differences: $dR = dR + 4$, $dG = dG + 4$
4: Encode as **b01**$||dR||dG - dR$, with dR and dG represented as 3-bit values
5: **Output:** Encoded data of 1 byte length

2.2.3. DIFF UP MODE **DIFF UP MODE** encodes the difference between the current pixel and the pixel above it. It operates similarly to DIFF MODE and is used when DIFF MODE is inapplicable, striving to encode a pixel within one byte. (See Algorithm 9.)

Algorithm 9 DIFF UP MODE.

1: **Input:** Pixel sequence to be encoded
2: Compute differences from the upper pixel: $dR1 = px_R - up_px_R$, $dG1 = px_G - up_px_G$
3: Adjust differences: $dR1 = dR1 + 4$, $dG1 = dG1 + 4$
4: Encode as **b00**$||dR1||dG1$, with $dR1$ and $dG1$ represented as 3-bit values
5: **Output:** Encoded data of 1 byte length

2.2.4. LUMA MODE **LUMA MODE** encodes pixels whose difference from the previous pixel is large. It exploits the property that the differences of the two channels relative to the first channel are often small, allowing compression of larger differences using two bytes. This mode is infrequently activated in typical pathology images. (See Algorithm 10.)

Algorithm 10 LUMA MODE.

1: **Input:** Pixel sequence to be encoded
2: Compute differences from the left pixel: $dR = px_R - left_px_R$, $dG_dR = (px_G - left_px_G) - dR$
3: Adjust differences: $dR = dR + 4$, $dG_dR = dG_dR + 4$
4: Encode as **b10**$||dR||dG_dR$, with both values represented as 3-bit values
5: **Output:** Encoded data of 1 byte length

2.2.5. RG MODE **RG MODE** encodes pixels when differences are too large for DIFF or LUMA MODE. It is the least frequently used mode in pathology images. (See Algorithm 11.)

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

Algorithm 11 RG MODE.

Input: Pixel to be encoded
Encode as **b11111111**$\|R\|G$
Output: Encoded data of 3 bytes length

Algorithm 12 Lossless compression for 8-bit two-channel pathology images.

Input: 8-bit two-channel image
Obtain image dimensions: height H, width W; initialize encoded array BM (each element $0-255$)
Initialize 12-byte header: $BM = header = [ASCII(QOI2), 4byte(H), 4byte(W)]$
Encode the first pixel using **RG MODE**: [**b11111111**, R, G]
for $h = 0, w = 1$ to $w = W - 1$ **do**
    Extend BM using QOI encoding along the first row
end for
for $h = 1, w = 0$ to $h = H - 1$ **do**
    Extend BM using QOI encoding along the first column
end for
for $h = 1, w = 1$ to $h = H - 1, w = W - 1$ **do**
    Compute differences with left pixel: $dR = R[h, w] - R[h, w-1]$, $dG = G[h, w] - G[h, w-1]$
    Compute differences with upper pixel: $dR1 = R[h, w] - R[h-1, w]$, $dG1 = G[h, w] - G[h-1, w]$
    if $dR == 0$ **and** $dG == 0$ **then**
        Extend BM using **RUN MODE**
    else if $-4 \le dR \le 3$ **and** $-4 \le dG \le 3$ **then**
        Extend BM using **DIFF MODE**
    else if $-4 \le dR1 \le 3$ **and** $-4 \le dG1 \le 3$ **then**
        Extend BM using **DIFF UP MODE**
    else if $-4 \le dR \le 3$ **and** $-4 \le dG - dR \le 3$ **then**
        Extend BM using **LUMA MODE**
    else
        Extend BM using **RG MODE**
    end if
end for
Output: Encoded array BM

2.2.6. Lossless compression algorithm for 8-bit two-channel pathology images The functionality of the QOI2 algorithm is illustrated in Figure 2. (See Algorithm 12.)

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

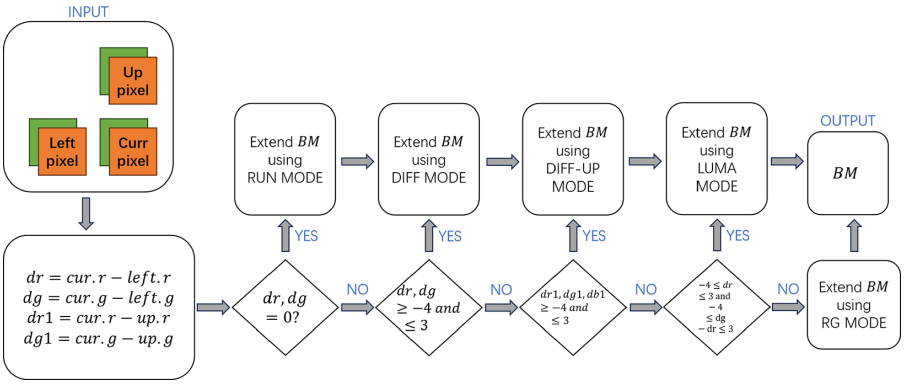


FIGURE 2. QOI2 lossless pathology images compression algorithm.

2.3. LOCO-I context encoding LOCO-I context encoding originates from the JPEG-LS image compression standard. This context-based predictive encoding estimates a pixel value using its neighbouring pixels to the left, top and top-left, and then encodes the resulting difference.

Applying LOCO-I encoding as a preprocessing step can further reduce spatial correlation, effectively "centring" pixel values and producing smaller residuals. When combined with our improved QOI algorithm, this preprocessing can lead to shorter encodings for certain images. The advantage arises because LOCO-I considers an additional top-left pixel, providing a slight spatial prediction improvement over our QOI-based encoding. (See Algorithm 13.)

Algorithm 13 Improved LOCO-I.

Input: Current pixel X and neighbouring pixels A (left), B (top), C (top-left)
if pixels A, B and C exist **then**
    if $C \geq \max(A, B)$ **then**
        $Pre = \min(A, B)$
    else if $C \leq \min(A, B)$ **then**
        $Pre = \max(A, B)$
    else
        $Pre = A + B - C$
    end if
else if only top pixel B does not exist **then**
    $Pre = A$
else
    $Pre = B$
end if
Output: Predicted pixel Pre and difference $Pre - X$

2.4. Lossless compression algorithm for 8-bit three-channel images (QOI3JPEGLS) We propose the QOI3JPEGLS lossless compression algorithm,

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

which integrates LOCO-I encoding with QOI3 encoding. The algorithm first applies LOCO-I encoding to the image, followed by QOI3 encoding. Due to the losslessness and reversibility of both LOCO-I and QOI3, the combined QOI3JPEGLS algorithm is also fully lossless and reversible.

2.5. Lossless compression algorithm for 8-bit two-channel images (QOI2JPEGLS) Similarly, we propose the QOI2JPEGLS algorithm, integrating LOCO-I encoding with QOI2 encoding. LOCO-I is applied first, followed by QOI2 encoding. The combination preserves the losslessness and reversibility of both components.

2.6. Parallelization algorithm Parallelization is critical for fully exploiting hardware performance. Many traditional image compression algorithms do not parallelize effectively: some cannot be parallelized at all, while others support only limited parallelism. Our proposed algorithms support parallelization at every stage, maximizing computational efficiency, which is particularly important for large pathology images containing millions of pixels.

Recent 2025 work also explores parallel-friendly designs: SEEC employs ROI-guided multi-entropy coding while maintaining decode independence across tiles/regions [24], and WISE demonstrates WSI-scale tiling pipelines with high throughput on CPUs/GPUs [9]. Our decoupled DIFF UP mode and pixel traversal order provide a codec-level alternative to such region-conditioned approaches, enabling end-to-end parallelism without external semantic information.

Prior to our improvements, LOCO-I context encoding, three-channel 8-bit QOI and two-channel 8-bit QOI did not fully support parallelization. To enhance parallelizability, we introduced the following modifications.

(1) Removing INDEX encoding to broaden contextual dependencies.
(2) Optimizing pixel traversal order.
(3) Introducing DIFF UP MODE to decouple inter-row pixel dependencies during encoding and decoding.

Tables 7 and 8 illustrate the pixel traversal orders for parallel encoding and decoding, respectively.

TABLE 7. Parallel encoding order (numbers indicate encoding sequence).

1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
1	2	3	4	5	6	7	8	9	10	⋯
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

TABLE 8. Parallel decoding order (numbers indicate decoding sequence).

0	0	0	0	0	0	0	0	0	0	⋯
0	1	2	3	4	5	6	7	8	9	⋯
0	2	3	4	5	6	7	8	9	10	⋯
0	3	4	5	6	7	8	9	10	11	⋯
0	4	5	6	7	8	9	10	11	12	⋯
0	5	6	7	8	9	10	11	12	13	⋯
0	6	7	8	9	10	11	12	13	14	⋯
0	7	8	9	10	11	12	13	14	15	⋯
0	8	9	10	11	12	13	14	15	16	⋯
0	9	10	11	12	13	14	15	16	17	⋯
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

3. Experimental results

3.1. Experimental design *Three-channel images:* We compare the average compression ratios of traditional QOI, PNG, QOI3 and QOI3JPEGLS across constructed datasets, including screenshots, pathology images and landscape photography. Additionally, representative compression results for individual images in each category are presented.

Two-channel pathology images: We evaluate the average compression ratios of PNG (per-channel compression), QOI2 and QOI2JPEGLS on pathology images, accompanied by representative data for individual images.

Parallel performance: The parallel speed performance of each algorithm for both three-channel and two-channel image compression is compared across different numbers of threads (n) on the constructed datasets. Trends in parallel processing speed and scalability for the proposed compression algorithms are also analysed.

3.2. Definitions and metrics

3.2.1. Original image size For 8-bit three-channel or two-channel images, the original image size is defined as

$$\text{Original image size} = \text{Number of Channels} \times \text{Height} \times \text{Width} \quad \text{(in bytes)}.$$

3.2.2. Compressed size (smaller is better) Defined as the total number of *encoded bytes* after compression.

3.2.3. Compression ratio (smaller is better) Defined as

$$\text{Compression Ratio} = \frac{\text{Compressed Size}}{\text{Original Image Size}} \times 100\%.$$

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

3.3. Comparison of three-channel images

TABLE 9. Compression ratios for three-channel screenshots.

Algorithm \ Images				
PNG	0.5371	0.4086	0.4416	0.3450
QOI	0.4927	0.3547	0.4682	0.3068
QOI3	0.4669	0.3250	**0.3978**	**0.2988**
QOI3JPEGLS	**0.4477**	**0.3244**	0.4274	0.3210

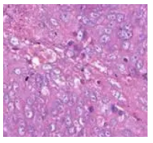
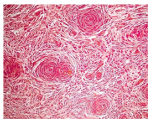
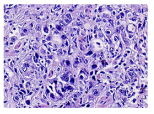
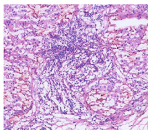
3.3.1. *Screenshots* (See Table 9.)

TABLE 10. Compression ratios for three-channel landscape images.

Algorithm \ Images				
PNG	0.5901	0.5890	0.6362	0.5396
QOI	0.5614	0.5169	0.5004	0.4529
QOI3	**0.4944**	**0.4845**	**0.4619**	**0.4140**
QOI3JPEGLS	0.5507	0.5388	0.4988	0.4510

3.3.2. *Landscape photography* (See Table 10.)

TABLE 11. Compression ratios for three-channel pathology images.

Algorithm \ Images				
PNG	0.8281	0.9410	0.9557	0.8995
QOI	0.6003	0.7930	0.9094	0.7446
QOI3	0.5449	0.7759	0.8961	0.7284
QOI3JPEGLS	**0.4313**	**0.6847**	**0.7766**	**0.6515**

3.3.3. *Three-channel pathology images* (See Table 11.)

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

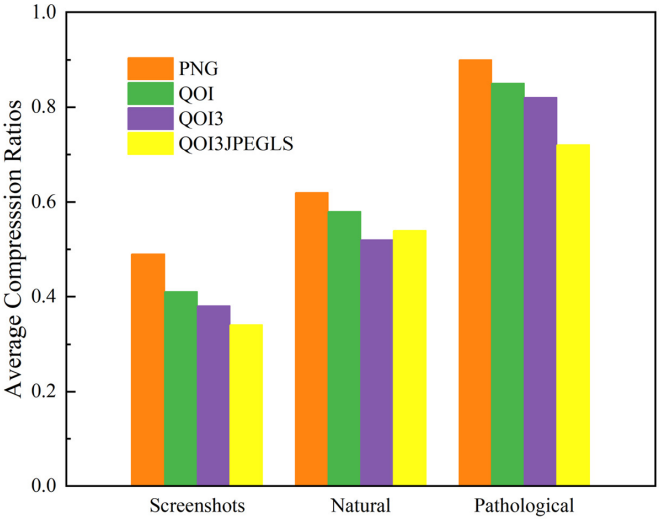
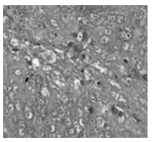
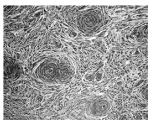
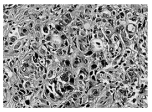
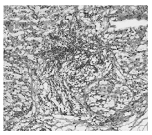


FIGURE 3. Average compression ratios for three-channel images.

3.3.4. Average compression ratios (See Figure 3.)

3.4. Comparison of two-channel pathology images (See Table 12.)

TABLE 12. Compression ratios for two-channel pathology images.

Algorithm \ Images				
PNG	0.5141	0.6980	0.7564	0.6955
QOI2	0.5218	0.8047	0.7982	0.7308
QOI2JPEGLS	**0.3580**	**0.6881**	**0.7357**	**0.6502**

3.4.1. Average compression ratios These ratios for two-channel images are illustrated in Figure 4.

3.5. Parallel speed comparison The parallel efficiency of the proposed method was evaluated by comparing the compression speed under different numbers of threads. The specific speedup ratios are presented in Table 13 and the corresponding performance trend is illustrated in Figure 5. The results demonstrate favourable parallel scalability for all methods, with the QOI2JPEGLS and QOI3JPEGLS hybrids achieving the highest parallel efficiency at 8 threads.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

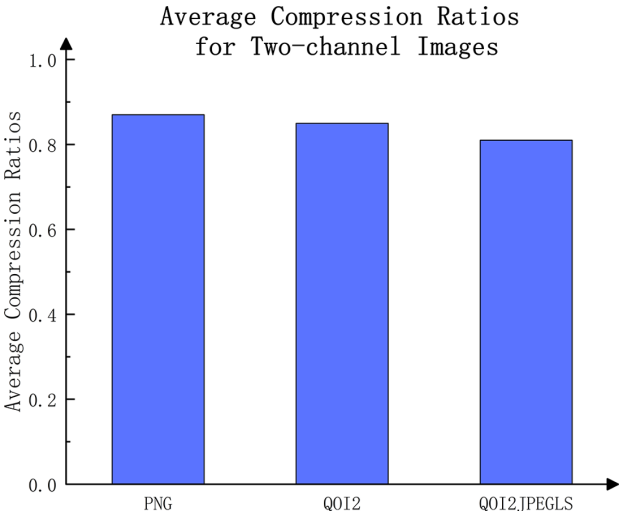


FIGURE 4. Average compression ratios for two-channel images.

TABLE 13. Parallel performance across number of threads (n).

n (Threads)	1	2	3	4	5	6	7	8
QOI3	1.00	0.59	0.41	0.32	0.30	0.29	0.28	0.28
QOI2	1.00	0.60	0.43	0.35	0.34	0.33	0.33	0.33
JPEGLS	1.00	0.55	0.40	0.30	0.24	0.22	0.20	0.20
QOI3JPEGLS	1.00	0.65	0.50	0.45	0.43	0.42	0.41	0.41
QOI2JPEGLS	1.00	0.67	0.52	0.48	0.46	0.45	0.43	0.43

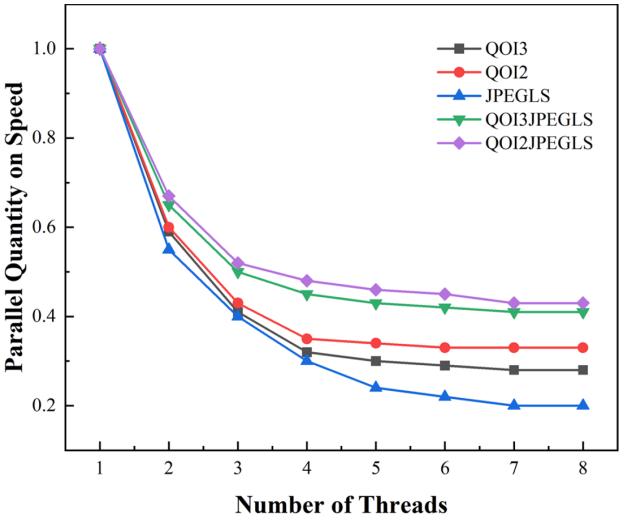


FIGURE 5. Impact of number of threads on compression speed.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

3.6. Discussion of experimental results The proposed QOI3 and QOI3JPEGLS algorithms consistently achieve superior compression ratios for both natural and pathology images compared with PNG and traditional QOI, although the advantage is less pronounced for screenshots. For two-channel pathology images, QOI2 and QOI2JPEGLS outperform PNG and per-channel compression approaches due to their consideration of inter-channel.

4. Conclusion

We have presented **QOI3/QOI3JPEGLS** (three-channel) and **QOI2/QOI2 JPEGLS** (two-channel) pathology codecs, which achieve superior lossless compression ratios on both natural and pathology images compared with PNG, baseline QOI and per-channel variants, while preserving the simplicity, determinism and end-to-end parallel scalability of QOI-style designs.

The proposed DIFF, DIFF UP and LUMA modes, together with the reordered traversal strategy, effectively eliminate serial dependencies and enable near-linear throughput scaling across CPU and GPU threads. When used with the reversible LOCO-I predictor, residual entropy is further reduced for specific content types without compromising parallelism or strict losslessness.

Future work will focus on enhancing the generality, efficiency and applicability of the codec family as follows.

(i) Bit-depth and channel generalization: extend encoding ranges and mode thresholds to higher bit-depth imagery (10/12/16-bit) and multi-channel formats (e.g. RGBA, multispectral/hyperspectral pathology) while maintaining reversible bitstreams.
(ii) Predictor and model exploration: investigate lightweight, parallelizable predictors beyond LOCO-I, such as CALIC-style adaptive contexts, to further reduce residual entropy without degrading decode parallelism.
(iii) Streaming and hardware acceleration: explore containerized bitstreams with chunked headers and per-tile restart markers for ROI decoding and low-latency streaming, alongside optimized SIMD/GPU/FPGA implementations to maximize practical throughput.

These directions aim to further improve the compression ratio–throughput trade-off, broaden applicability across medical and graphics domains, and ensure openness and reproducibility for future research and deployment.

Acknowledgements

This research was supported by the National Key Research and Development Program of China, Grant Number 2020YFA0712500; and the Research and Development Project of Pazhou Lab (Huangpu), Grant Number 2023K0601.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

References

[1] T. Boutell, *PNG (portable network graphics) specification version 1.0* (1997); https://www.rfc-editor.org/rfc/rfc2083.

[2] C. de Cea-Dominguez, J. C. Moure, J. Bartrina-Rapesta and F. Aulí-Llinàs, "Real-time 16k video coding on a GPU with complexity scalable BPC-PaCo", *Signal Process. Image Commun.* **99** (2021) article ID: 116503; doi:10.1016/j.image.2021.116503.

[3] C. De Cea-Dominguez, J. C. Moure-Lopez, J. Bartrina-Rapesta and F. Aulí-Llinàs, "GPU-oriented architecture for an end-to-end image/video codec based on JPEG2000", *IEEE Access* **8** (2020) 68474–68487; doi:10.1109/ACCESS.2020.2985859.

[4] J. Du, C. Zhou, N. Cao, G. Chen, Y. Chen, Z. Cheng, L. Song, G. Lu and W. Zhang, "Large language model for lossless image compression with visual prompts", Preprint, 2025, arXiv:2502.16163.

[5] D. A. Huffman, "A method for the construction of minimum-redundancy codes", *Proc. IRE* **40** (2007) 1098–1101; doi:10.1109/JRPROC.1952.273898.

[6] D. Li, Y. Bai, K. Wang, J. Jiang, X. Liu and W. Gao, "CALLIC: content adaptive learning for lossless image compression", in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 39 (eds T. Walsh, J. Shah and Z. Kolter) (AAAI Press, Philadelphia, PA, 2025) 4679–4688; doi:10.1609/aaai.v39i5.32494.

[7] Z. Li, C. Huang, X. Wang, H. Hu, C. Wyeth, D. Bu, Q. Yu, W. Gao, X. Liu and M. Li, "Lossless data compression by large models", *Nat. Mach. Intell.* **7** (2025) 794–799; doi:10.1038/s42256-025-01033-7.

[8] H. Liao and Y. Li, "LFC-UNet: learned lossless medical image fast compression with U-Net", *PeerJ. Comput. Sci.* **10** (2024) article ID: e1924; doi:10.7717/peerj-cs.1924.

[9] Y. Mao, J. Wang, N. Guan and C. J. Xue, "WISE: a framework for gigapixel whole-slide-image lossless compression", in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, 2025 (IEEE Computer Vision Foundation, 2025) 29342–29351; doi:10.1109/CVPR52734.2025.02732.

[10] M. A. Rahman and M. Hamada, "Lossless image compression techniques: a state-of-the-art survey", *Symmetry* **11** (2019) article ID: 1274; doi:10.3390/sym11101274.

[11] H. Rhee, Y. I. Jang, S. Kim and N. I. Cho, "LC-FDNet: learned lossless image compression with frequency decomposition network", in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, 2022 (IEEE, 2022) 6033–6042; doi:10.1109/CVPR52688.2022.00594.

[12] J. J. Rissanen, "Generalized Kraft inequality and arithmetic coding", *IBM J. Res. Dev.* **20** (1976) 198–203; doi:10.1147/rd.203.0198.

[13] I. Schiopu and A. Munteanu, "Deep-learning-based lossless image coding", *IEEE Trans. Circuits Syst. Video Technol.* **30** (2020) 1829–1842; doi:10.1109/TCSVT.2019.2909821.

[14] Z. Shi, W. Ma, G. Wu, Y. Du, M. Xia and Z. Li, "Design of tile-based ARGB image lossless compressor and decompressor", in: *2023 IEEE 17th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, Xiamen, China, 2023 (IEEE, 2023) 57–61; doi:10.1109/ASID60355.2023.10426160.

[15] A. Skodras, C. Christopoulos and T. Ebrahimi, "The JPEG 2000 still image compression standard", *IEEE Signal Process. Mag.* **18** (2002) 36–58; doi: 10.1109/79.952804.

[16] J. Sneyers and P. Wuille, "FLIF: free lossless image format based on MANIAC compression", in: *2016 IEEE International Conference on Image Processing (ICIP 2016)*, Phoenix, AZ, 2016 (IEEE, 2016) 66–70; doi:10.1109/ICIP.2016.7532320.

[17] D. Szablewski, "QOI – the quite OK image format for fast, lossless compression" (2022); https://qoiformat.org/.

[18] D. S. Taubman, M. W. Marcellin and M. Rabbani, "JPEG2000: image compression fundamentals, standards and practice", *J. Electron. Imaging* **11** (2002) 286–287; doi:10.1117/1.1469618.

[19] G. K. Wallace, "The JPEG still picture compression standard", *Commun. ACM* **34** (1991) 30–44; doi: 10.1145/103085.103089.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press

[20] M. J. Weinberger, G. Seroussi and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS", *IEEE Trans. Image Process.* **9** (2000) 1309–1324; doi:10.1109/83.855427.

[21] X. Wu and N. Memon, "CALIC – a context based adaptive lossless image codec", in: *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Atlanta, GA, 1996, Volume 4 (IEEE Computer Society, 1996) 1890–1893; doi:10.1109/ICASSP.1996.544819.

[22] X. Wu and N. Memon, "Context-based lossless interband compression-extending CALIC", *IEEE Trans. Image Process.* **9** (2000) 994–1001; doi:10.1109/83.846242.

[23] Z. Zhang, Z. Chen and S. Liu, "Fitted neural lossless image compression", in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, 2025 (IEEE Computer Vision Foundation, 2025) 23249–23258; doi:10.1109/CVPR52734.2025.02165.

[24] C. Zheng, Z. Ren and D. Li, "SEEC: segmentation-assisted multi-entropy models for learned lossless image compression", Preprint, 2025, arXiv:2509.07704.

https://doi.org/10.1017/S1446181125100242 Published online by Cambridge University Press