CMPE362 Project 4

Yusuf AKIN, 2021400288; Ahmet Salih TURKEL, 2021400120

June 1, 2025

1 Implementation

In Part 1, we implemented the simple encoder/decoder pipeline as described in the project specification. We omit low-level serialization/deserialization details; the high-level data flow is:

Compression Pipeline:

- Read input image.
- Compute frame differences (for P-frames).
- Divide each frame into 8×8 blocks.
- Apply 2D DCT to each block.
- Divide DCT coefficients by the quantization matrix (element-wise).
- Perform zigzag scanning on each block.
- Apply run-length encoding (RLE) to the zigzag vectors.
- Serialize the RLE outputs into a binary stream.

Decompression Pipeline:

- Read the binary stream.
- Inverse RLE on each block.
- Inverse zigzag scan to reconstruct each 8×8 coefficient block.
- Multiply each block by the quantization matrix (element-wise).
- Apply inverse 2D DCT to each block.
- Reassemble all blocks into full-frame images.
- Recover original frames by adding frame differences (for P-frames).

During serialization, each color channel's RLE output is stored as an $(X_i \times 2)$ matrix of `int8` per frame. Before writing those three matrices (R, G, B) back-to-back, we prepend metadata to the binary file in this exact order:

1. GOP size
2. Quantization-matrix dimensions and entries

3. Number of images
4. Image dimensions
5. Sizes of each RLE block per channel
6. Concatenated RLE matrices (R, G, B)

In the decompression step, we read the metadata in the same sequence, reconstruct each channel's RLE blocks, then apply the inverse pipeline above to recover the original JPEG frames.

2 GOP Size vs Compression Ratio

Varying the GOP size has a significant effect on compression efficiency: larger GOP sizes allow more inter-frame prediction (more P-frames per I-frame) and therefore higher overall compression. However, as Figure 1 shows, the gain diminishes once the GOP size exceeds approximately 12–15. In practice, GOP sizes larger than 15 yield negligible additional compression, so using GOP 15 is not recommended unless extreme file-size reduction outweighs decoding complexity.

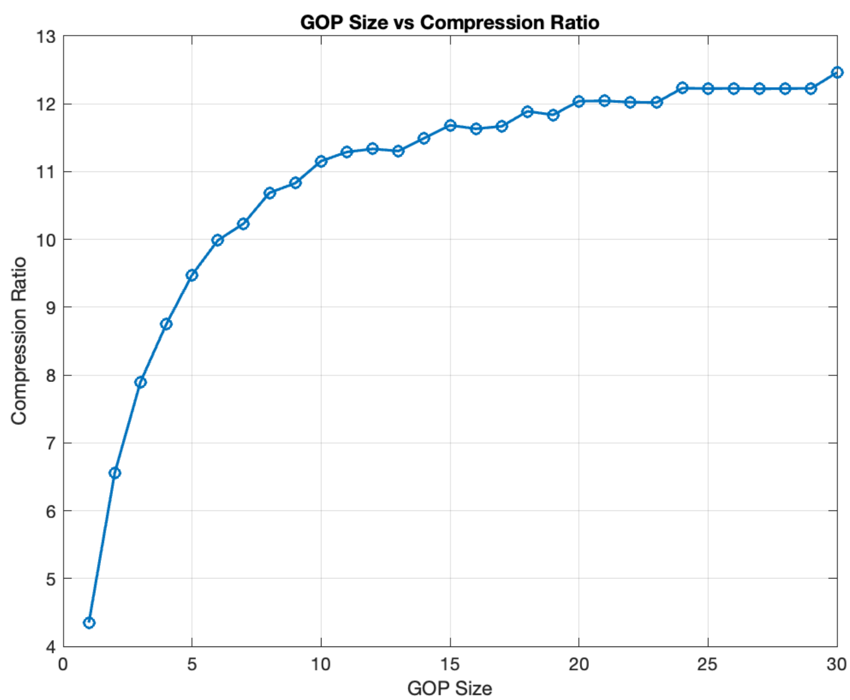

Figure 1: Compression Ratio versus GOP Size (Part 1 Encoder). Compression ratio improves rapidly up to GOP ≈12, then plateaus beyond GOP 15.

3 PSNR Analysis

Figure 2 plots PSNR (dB) against frame index for three GOP configurations: GOP = 1 (all I-frames, red), GOP = 15 (green), and GOP = 30 (blue). As the GOP size increases, PSNR in the middle of each GOP drops more severely due to error accumulation in consecutive P-frames. Therefore, smaller GOP sizes yield higher fidelity (higher average PSNR), at the cost of lower compression. If preserving image fidelity is critical, avoid large GOP sizes.

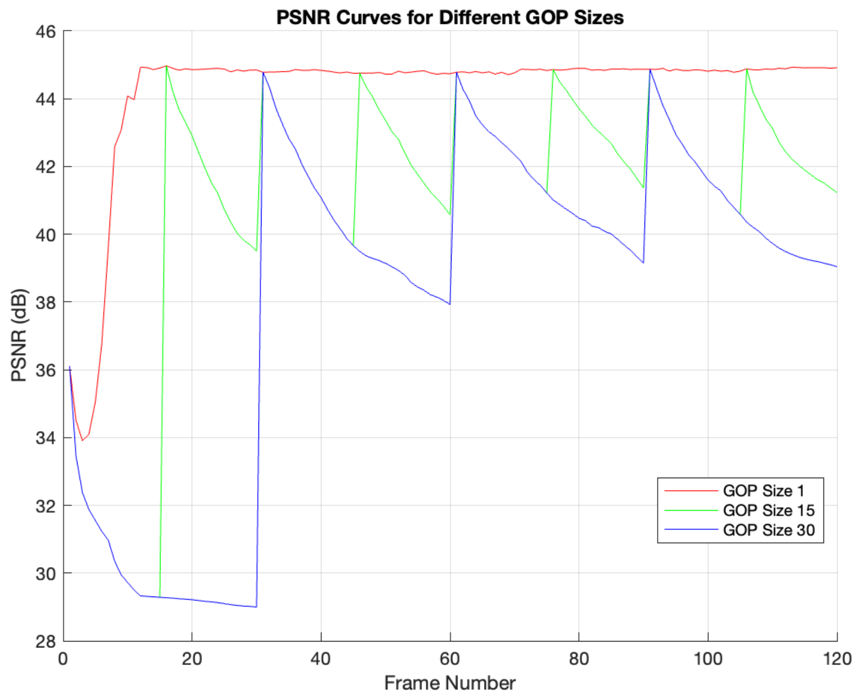

Figure 2: Frame-by-Frame PSNR for GOP Sizes 1, 15, 30 (Part 1 Encoder). Larger GOPs exhibit lower PSNR mid-GOP due to cumulative inter-frame error.

4 Improved Compression Algorithm with B-Frames

4.1 Algorithm Description

We extend the Part 1 encoder by including B-frames with temporal prediction. First, we generate a GOP layout using three parameters: total frame count, GOP size, and the number of B-frames between consecutive anchor frames (I or P). The steps are as follows:

1. **I-frames**: Encode each I-frame independently using 2D DCT, quantization, zigzag, and RLE, exactly as in Part 1.
2. **P-frames**: Predict each P-frame from the *reconstructed* version of its preceding anchor. In contrast, Part 1 predicted from the original frame; here we predict from the decoded (reconstructed) reference so that the encoder and decoder see the same prediction input.
3. **B-frames**: Given a B-frame at index t, let t_{prev} and t_{next} be the indices of its nearest decoded

anchors (I or P) before and after it. Compute the temporal weight

$$\alpha = \frac{t - t_{\text{prev}}}{t_{\text{next}} - t_{\text{prev}}} \quad \text{(ranges from 0 to 1).}$$

Then form a *prediction image* by linear interpolation of the *reconstructed* anchor frames:

$$\widehat{B}(x, y) = (1 - \alpha)\, B_{\text{prev}}(x, y) + \alpha\, B_{\text{next}}(x, y) \quad \forall\, \text{pixel } (x, y).$$

Subtract this predicted macroblock from the original macroblock to obtain the residual, then apply 2D DCT, quantization, zigzag, and RLE to that residual. During decoding, the inverse pipeline reconstructs the residual and adds it back to the same weighted interpolation of the decoded anchors.

4.2 Compression Ratio Results

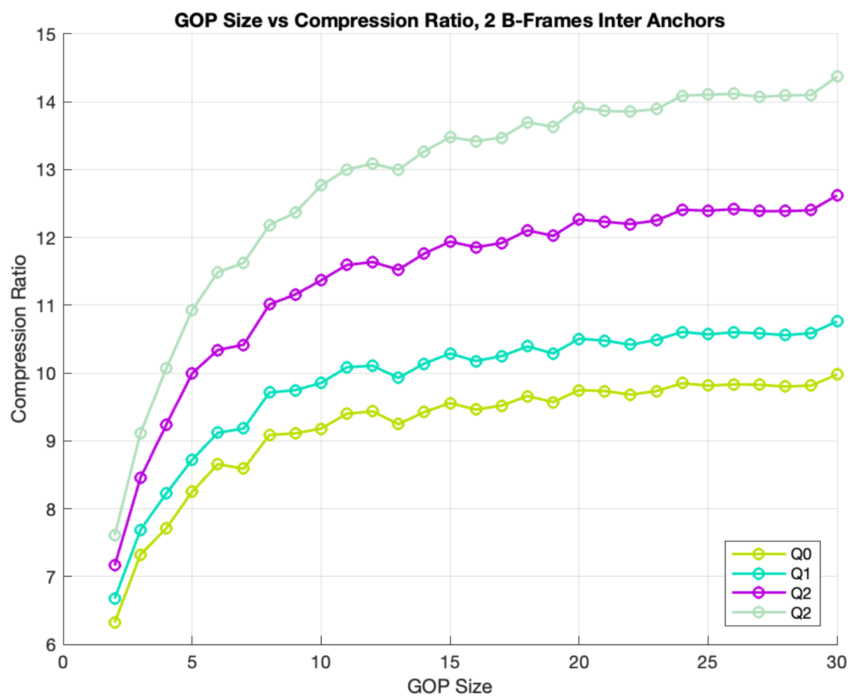

Figure 3: Compression Ratio vs. GOP Size for Quantization Matrices $Q_0 \ldots, Q_3$ with 2 B-Frames per GOP. Coarser Q_i yields higher compression at the expense of quality.

In the improved encoder, we also experimented with four quantization matrices:

$$Q_0 = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}, \quad Q_1 = \begin{bmatrix} 3 & 5 & 7 & 9 & 11 & 13 & 15 & 17 \\ 5 & 7 & 9 & 11 & 13 & 15 & 17 & 19 \\ 7 & 9 & 11 & 13 & 15 & 17 & 19 & 21 \\ 9 & 11 & 13 & 15 & 17 & 19 & 21 & 23 \\ 11 & 13 & 15 & 17 & 19 & 21 & 23 & 25 \\ 13 & 15 & 17 & 19 & 21 & 23 & 25 & 27 \\ 15 & 17 & 19 & 21 & 23 & 25 & 27 & 29 \\ 17 & 19 & 21 & 23 & 25 & 27 & 29 & 31 \end{bmatrix},$$

$$Q_2 = 3\, Q_1, \qquad Q_3 = 5\, Q_1.$$

Figure 3 shows the compression ratio vs. GOP size for each Q_i when using 2 B-frames between anchors. As quantization becomes coarser (from Q_0 to Q_3), compression ratio increases (more bits saved) at the cost of lower PSNR. In all cases, the compression gain from increasing the GOP size diminishes once the GOP exceeds roughly 12–15.

Figure 4 fixes GOP = 30, compares compression ratio as the number of B-frames varies (quantization Q_0). As the count of B-frames increases, compression ratio decreases: including more B-frames (with their interpolation residuals) adds overhead compared to a pure I/P structure.

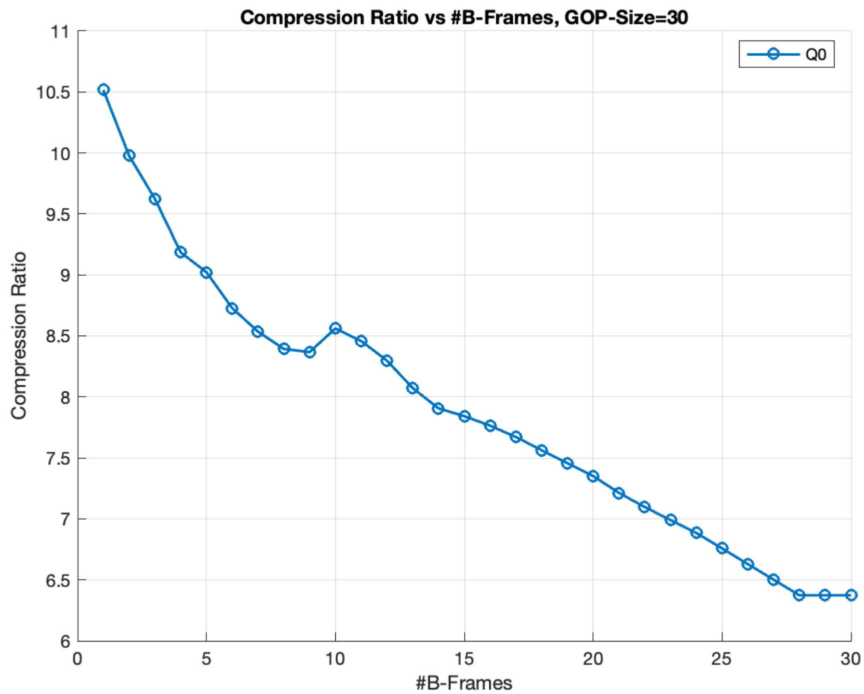

Figure 4: Compression Ratio vs. #B-Frames (GOP = 30, Quantization Q_0). More B-frames reduce compression efficiency.

4.3 PSNR Analysis

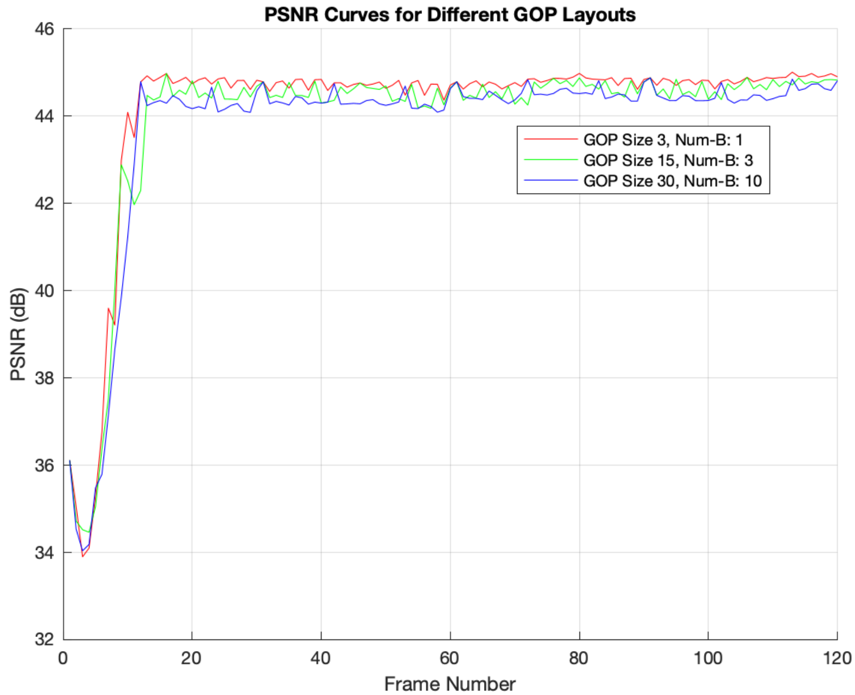

Figure 5: Frame-by-Frame PSNR for Different GOP Layouts with B-Frames (Quantization Q_0). B-frames significantly reduce PSNR fluctuation compared to I/P-only.

Figure 5 plots frame-by-frame PSNR for three encoder variants:

- **GOP 3, 1 B-frame** (red),
- **GOP 15, 3 B-frames** (green),
- **GOP 30, 10 B-frames** (blue).

Thanks to B-frame interpolation, the PSNR curve is much flatter across frames: prediction error is shared between two anchors, so no frame experiences the severe quality dip seen in the I/P-only encoder (compare to Figure 2). This confirms that B-frames smooth out visual quality (at modest extra bit overhead).

Figure 6 compares PSNR for the same GOP layout (GOP = 15, 2 B-frames) under three quantization settings:

- **Quantization** Q_0 (red),
- **Quantization** Q_1 (green),
- **Quantization** Q_2 (blue).

With the finest quantization Q_0, PSNR settles around 44–45 dB after the initial frames, indicating high-fidelity reconstruction. Switching to the coarser Q_1 matrix dramatically lowers PSNR to approximately 17–18 dB, reflecting substantial quality degradation. The intermediate Q_2 case

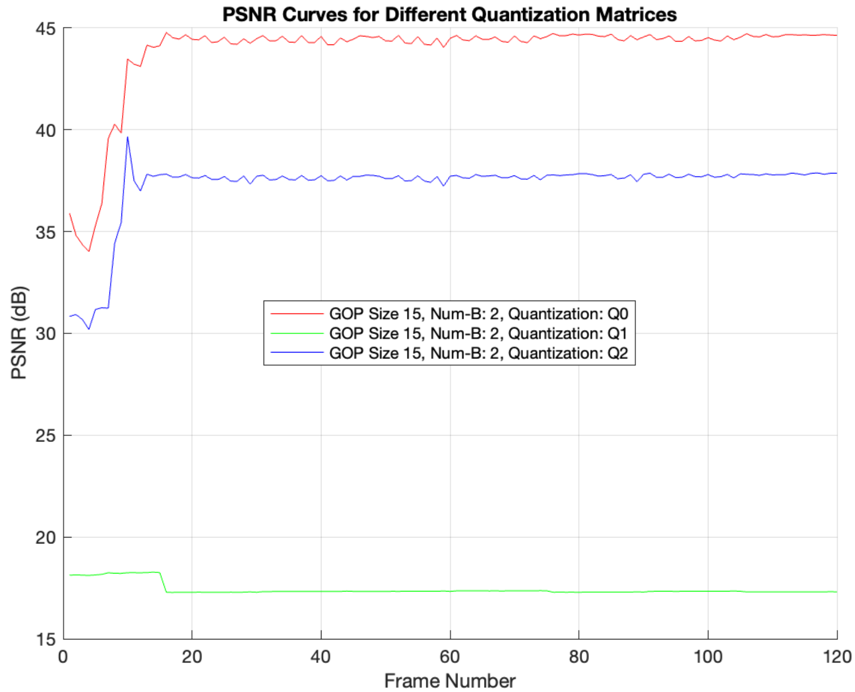

Figure 6: Frame-by-Frame PSNR for GOP Size 15 with 2 B-Frames, using Quantization Matrices Q_0, Q_1, and Q_2.

yields a mid-range PSNR of about 37–38 dB. These curves illustrate the expected trade-off: as quantization coarsens, bit-rate decreases but reconstruction quality (PSNR) falls significantly.