Optimizing the Performance and Efficiency of a Hybrid Video Encoder in Java

Maciej Okoń[1][0009-0005-7427-6784] and Beata Bylina[1][0000-0002-1327-9747]

Maria Curie-Skłodowska University, Plac Marii Curie-Skłodowskiej 5, 20-031 Lublin, Poland
maciej.okon@mail.umcs.pl, beata.bylina@mail.umcs.pl

Abstract. This paper explores the architecture of a hybrid video encoder implemented in Java, addressing the challenges of high-performance multimedia processing within a managed environment. The study focuses on balancing algorithmic complexity with execution efficiency, specifically managing temporal dependencies and memory overhead inherent in multi-threaded video compression.
The proposed architecture incorporates four key optimization mechanisms: a complete inter-frame processing pipeline, motion estimation with Half-Pel precision, a P-SKIP mode, and entropy coding. To maximize throughput, the encoder utilizes a Group of Pictures (GOP)-parallel model, where multiple threads independently process frame sequences of up to 30 frames.
Experimental results demonstrate bitrate reductions reaching up to 48% compared to the baseline intra-only version while achieving a structural similarity index (SSIM) exceeding 0.95. The implementation demonstrates that GOP-level parallelism in Java effectively maintains high encoding throughput and stability across diverse many-core architectures. The study confirms that combining sub-pixel estimation with adaptive complexity management optimizes Rate-Distortion performance while effectively minimizing computational overhead.

Keywords: Video Compression · GOP-level Parallelism · Rate-Distortion Performance · Computational Optimization · Motion Estimation · Java

1 Introduction

The increasing demand for efficient video stream processing, driven by the rapid growth of cloud services and streaming platforms, presents significant computational optimization challenges for contemporary computing architectures. In distributed environments, where code flexibility and portability are key, solutions based on high-level managed languages, such as Java, are gaining importance. Traditionally, however, Java has been perceived as unsuitable for tasks requiring intensive pixel-level calculations due to the overhead associated with memory management (Garbage Collection) and the lack of direct control over processor instructions, which necessitates innovative approaches to algorithmic efficiency and resource utilization.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

In our previous research [1], we addressed this challenge by proposing a multi-threaded Java-based video encoder utilizing a static frame-level parallelism model. This work confirmed that achieving high throughput on multi-core systems is possible within the JVM environment. However, the initial implementation prioritized raw processing speed at the expense of compression efficiency, representing a suboptimal trade-off in the Rate-Distortion-Complexity space. It relied on a simplified coding pipeline limited exclusively to Intra-frame compression, treating the video sequence as a series of independent images and ignoring temporal correlation. Additionally, it used direct binary output of quantized coefficients, omitting the entropy coding stage, which is crucial for efficient data reduction. As a result, the generated bitstreams were characterized by a suboptimal ratio of quality to file size.

This paper treats the development of a hybrid encoder as an optimization problem, balancing compression efficiency (Rate-Distortion performance) against computational cost. Unlike our previous work, the introduction of inter-frame prediction necessitates a transition to a GOP-parallel (Group of Pictures) model to respect temporal dependencies while maintaining high CPU utilization. The primary objective of this work is to demonstrate that by employing targeted algorithmic optimizations and Half-Pel precision, it is possible to develop a Java-based encoder that effectively balances predictive accuracy with parallel efficiency on multi-core architectures.

The main contributions of this work are as follows:

1. **Scalable GOP-Parallel Optimization Model:** A task-decomposition strategy that ensures temporal consistency while maximizing multi-core utilization through independent segment processing.
2. **Numerical Efficiency in Motion Estimation:** Implementation of a hierarchical Half-Pel search algorithm using bit-accurate bilinear interpolation to minimize residual energy and bitstream volume.
3. **Adaptive Complexity Management:** Integration of a P-SKIP heuristic and scene change detection to optimize the trade-off between prediction accuracy and computational overhead.
4. **Empirical Scalability Analysis:** A comprehensive performance evaluation on many-core Intel Xeon clusters.

The paper is organized as follows. Section II discusses the complete inter-frame pipeline and the GOP-parallel architecture. Section III presents motion estimation with Half-Pel precision. Section IV describes temporal and computational redundancy reduction through the P-SKIP mode and adaptive GOP control. Section V details the implementation of the entropy coding module. Section VI provides a performance and quality analysis based on tests conducted on a server cluster. Finally, Section VII presents the conclusions.

2 GOP-Parallel Hybrid Architecture

Transitioning from a purely Intra-frame model to a hybrid model, combining intra-frame and inter-frame compression, requires a revision of parallelization

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

strategies. In previous work on I-VOP (Intra-coded Video Object Plane) coding, high throughput was achieved using frame-level parallelism, where multiple frames are processed simultaneously. However, the introduction of P-VOPs(Predicted Video Object Planes) enforces frame-by-frame processing: motion estimation for frame (N) cannot begin until the reference frame (N-1) is fully reconstructed. To respect these dependencies while maximizing CPU utilization, a Group of Pictures (GOP)-parallel architecture was implemented [4]. The architectural design of the proposed system is depicted in Fig. 1.

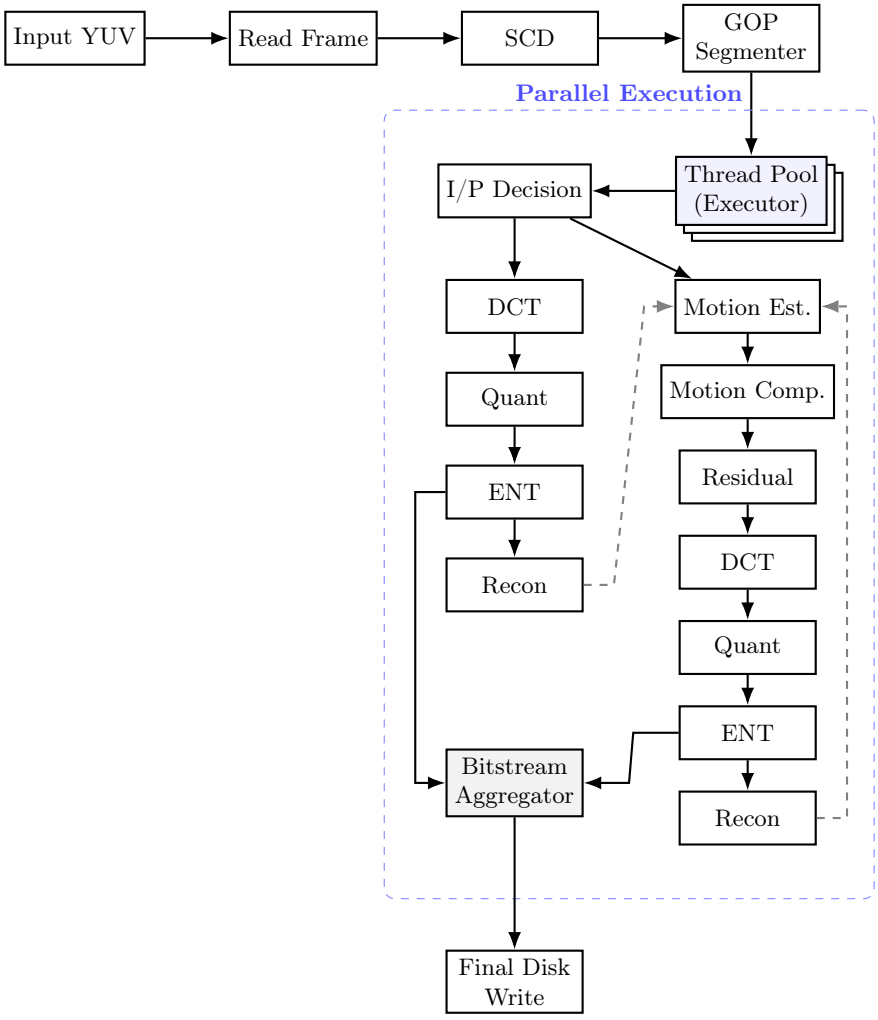

Fig. 1. System architecture highlighting the separation between the parallelized hybrid encoding process (GOP-level) and sequential I/O operations.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

2.1 Asynchronous GOP Processing Model

In the proposed solution, the video stream is divided into independent segments called GOPs. Each GOP, consisting of up to 30 frames, is treated as a single atomic task submitted to a thread pool managed by the `ExecutorService`. This gives two key advantages in Java:

1. **Dependency isolation:** Frames within a thread execute sequentially, allowing the encoder to keep a local reconstructed reference frame (`refReconYUV`) without the need for complex inter-thread synchronization or locking mechanisms.
2. **Order Preservation:** The main thread collects `Future<byte[]>` results, ensuring that even if GOPs finish out of order, the final bitstream is written to disk in chronological sequence.

2.2 Data Handling

The encoder bypasses Java's `BufferedImage` abstraction [2] and operates directly on a custom `YUVFrame` structure:

- **Raw Data Access:** Frames are read directly from the source file into primitive `byte[]` arrays representing the Y, U, and V planes.
- **Deterministic Memory Access:** The predictable, contiguous memory layout ensures high data locality and minimizes access latency during Sum of Absolute Differences (SAD) computations [3].

3 Motion Estimation

With the introduction of inter-frame prediction, temporal redundancy can be exploited directly rather than treating frames as independent images. This shift makes motion estimation a central component of the encoding pipeline, as prediction accuracy now directly determines residual energy and bitrate.

To achieve this, the encoding pipeline incorporates an inter-frame prediction stage. Instead of relying on integer-pixel block matching, the system utilizes Half-Pel (Half Picture Element) precision. This choice improves alignment between predicted and current macroblocks, especially in scenes involving slow motion or subtle spatial shifts, leading to lower residuals compared to integer-only estimation.

3.1 Half-Pel Motion Estimation Algorithm

Motion compensation requires an efficient search strategy that balances accuracy and computational cost. The `MotionVector` module applies a two-stage hierarchical procedure, combining a fast coarse search with local sub-pixel refinement:

1. **Integer-Pel search:** A preliminary motion vector is estimated using a Step Search algorithm [7] on the integer pixel grid. This stage determines the dominant displacement of the macroblock while keeping the search cost low.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

2. **Half-pixel refinement:** The best candidate from the first stage is refined by evaluating eight neighboring positions at a distance of ± 0.5 pixels. Since these sub-pixel positions are not present in the original sampling grid, they are generated on-the-fly using **bilinear interpolation**. The search selects the Half-Pel vector that minimizes the Sum of Absolute Differences (SAD) more effectively than the integer-pixel grid alone.

3.2 Bilinear Interpolation

Sub-pixel prediction requires estimating values between discrete samples. High-level image abstractions introduce unnecessary overhead in tight encoding loops, therefore interpolation is performed directly on primitive arrays.

A lightweight bilinear interpolation scheme operating on `int[][]` buffers is used. Instead of floating-point arithmetic, integer operations with bitwise shifts are applied for averaging. For example, a horizontal Half-Pel sample $P_{0.5,0}$ between pixels A and B is computed as:

$$P_{0.5,0} = (A + B + 1) \gg 1 \tag{1}$$

The central Half-Pel position interpolated from four neighboring pixels A, B, C, D is calculated as:

$$P_{0.5,0.5} = (A + B + C + D + 2) \gg 2 \tag{2}$$

This integer-based approach maintains a low memory footprint and enables the JVM to perform aggressive JIT optimizations, such as loop unrolling and vectorization [3].

3.3 Impact on Compression

Half-Pel motion estimation increases computational cost compared to integer-only prediction. This additional cost is offset by more accurate motion matching, which lowers the energy of the residual signal. As a result, fewer bits are required to encode prediction errors, improving rate-distortion (R-D) performance relative to integer-pixel estimation.

4 Temporal and Computational Redundancy Reduction

Two adaptive mechanisms were added to reduce unnecessary data processing, optimize bit allocation and align the encoder with modern compression efficiency standards. These mechanisms adjust the encoding process to handle static background regions and abrupt scene changes more efficiently.

In typical video sequences significant portions of the frame remain static over time. Standard P-VOP coding may still produce residual data in these regions due to signal noise, consuming bandwidth for motion vector headers and coefficients.

To mitigate this P-SKIP decision logic was integrated directly into the encoding loop. For each macroblock, the encoder evaluates two conditions:

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

1. **Zero Motion:** The best matching block in the reference frame is at the same spatial position (motion vector $(0, 0)$).
2. **Low Residual Energy:** SAD between the current block and the reference block falls below a perceptually tuned threshold.

When both conditions are satisfied, the block is marked as "skipped". In the bitstream, this is represented by a SKIP flag, instructing the decoder to copy the pixel data for the block from the previous frame.

4.1 Adaptive GOP and Scene Change Detection

The adopted hybrid model, operating on Groups of Pictures, introduces an additional challenge: sudden scene changes within a sequence of predicted frames. In such cases, the current frame may have little correlation with its reference frame, and a rigid GOP structure can produce visual artifacts and inefficient use of bandwidth.

To address this, instead of relying on a fixed keyframe interval, an adaptive mechanism based on a lightweight Scene Change Detection (SCD) module is used. Before encoding each frame, the system performs a quick pre-pass on a subsampled grid, inspecting every 8^{th} pixel.

When the mean pixel difference exceeds a dynamically determined threshold, a scene cut is detected. The current frame is then encoded as an I-VOP, effectively resetting the prediction chain. This ensures that each new scene starts with a high-quality, independent frame, preventing prediction errors from propagating from previous frames.

5 Entropy Coding Integration

The baseline encoder in [1] relied solely on Run-Length Encoding (RLE) for quantized DCT coefficients. While computationally cheap, this approach did not exploit the statistical distribution of video data, where small values dominate and long runs of zeros frequently occur.

To improve compression, the encoder was extended with a full entropy coding stage, combining DPCM, RLE, and Huffman coding. The `RLEHuffmanEncoder` module implements this process as follows.

5.1 Data Reordering and DPCM

Each input block is an 8×8 matrix of quantized DCT coefficients. The first element, the DC coefficient at position $[0, 0]$, represents the block's average brightness. Adjacent blocks typically have similar DC values, so we encode the difference from the previous block using Differential Pulse Code Modulation (DPCM). This produces small values centered around zero, which require fewer bits.

The remaining 63 AC coefficients represent spatial details. A Zig-Zag scan flattens the 2D block into a 1D array, clustering non-zero values at the start and leaving long runs of zeros at the end, making RLE effective.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

5.2 Hybrid RLE and Huffman Algorithm

The AC coefficient stream is converted into (Run, Level) pairs, where *Run* counts skipped zeros and *Level* is the coefficient value. To avoid creating inefficient code tables for all possible numerical values, the implemented algorithm does not encode the *Level* parameter directly. Instead, this value is decomposed into two components: *Category* (determining the order of magnitude) and *Amplitude* (refining the value). The bitstream writing process is performed in two steps:

1. **Huffman Coding:** First, the *Category* is determined, i.e., the number of bits necessary to store the absolute value of the coefficient. Next, a Huffman code [6] assigned to the pair (Run, Category) is retrieved from a static table. This code uniquely identifies the occurrence of a specific combination of zero count and coefficient magnitude.
2. **Amplitude Bits:** Immediately after the Huffman code, raw value bits (amplitude) are appended to the stream. Their count is strictly determined by the previously assigned Category. These bits allow for the precise distinction of a specific value within a given category.

Special symbols, such as `EOB` (End of Block) and `ZRL` (Zero Run Length), signal trailing zeros and long zero runs, allowing efficient termination of block encoding.

6 Performance Analysis and Results

6.1 Evaluation Methodology and Test Environment

To comprehensively verify the introduced optimizations, a multidimensional set of evaluation criteria was adopted, covering both processing performance and compression efficiency. Tests were conducted using a standard test sequence with a resolution of 1920×1080 pixels (Full HD) and a frame rate of 60 FPS (Frames per Second), in YUV 4:2:0 format. The specific sizes of the raw input data processed during the experiments are detailed in Table 1.

Table 1. Input data volume: Raw size (MB) for varying frame counts (1080p, YUV 4:2:0).

Frame Count	250	500	750	1000	1250	1500	1750	2000
Raw Size [MB]	742	1483	2225	2966	3708	4450	5191	5933

The following metrics are used consistently throughout the experimental section to compare different encoder configurations, including baseline and optimized variants. They allow for a joint assessment of reconstruction distortion, perceptual quality, and bitrate efficiency, where the bitrate is defined as the average size of the produced bitstream per unit of time.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

1. **Objective Quality (PSNR):** Peak Signal-to-Noise Ratio is a standard metric for assessing signal reconstruction fidelity [5]. The value for the luminance component (Y) is calculated according to the formula:

$$PSNR = 10 \cdot \log_{10} \left(\frac{(2^B - 1)^2}{MSE} \right) \tag{3}$$

   Where:
   - MSE is the Mean Squared Error between the pixels of the original and reconstructed images,
   - B is the bit depth of the sample (for the tested video format $B = 8$, resulting in a maximum signal value of 255).

   Higher PSNR values indicate a lower level of distortion between the original and reconstructed signal. Since PSNR is expressed on a logarithmic scale, each increase of 3 dB corresponds approximately to halving the mean squared error. In practice, low PSNR values reflect strong visible degradation, while higher values indicate that compression artifacts are weak and the reconstructed frame closely matches the original.
2. **Perceptual Quality (SSIM):** Structural Similarity Index [5]. It serves to evaluate visible image distortions by analyzing the correlation between local pixel patterns. The measure's value for two image windows x and y is defined as follows:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4}$$

   Where:
   - μ_x, μ_y are the average luminance values in windows x and y,
   - σ_x^2, σ_y^2 are the variances (representing contrast),
   - σ_{xy} is the covariance (representing structural similarity),
   - C_1, C_2 are stabilizing constants to prevent division by zero.

   The SSIM index takes values in the range $[-1, 1]$, where values close to 1 indicate a high degree of structural similarity between the original and reconstructed images. Unlike PSNR, SSIM emphasizes the preservation of local structures, contrast, and luminance relationships, which makes it more aligned with human visual perception.
3. **Rate-Distortion (R-D) Analysis:** A plot of image quality (Y-axis) versus the resulting output stream size (Bitrate, X-axis), allowing for the assessment of compression efficiency resulting from the application of more precise motion estimation. In R-D analysis, a curve that achieves higher quality at the same bitrate, or the same quality at a lower bitrate, is considered superior.

Performance evaluations were conducted on four server platforms featuring various generations of Intel Xeon processors (C1-C4), as detailed in Section 6.2. Due to the deterministic nature of the implemented parallel algorithm, the quality metrics (PSNR, SSIM, Bitrate) remain consistent across all hardware platforms. Consequently, the quality results section presents data independent of the specific processor architecture.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

6.2 Test Platforms

The experiments were conducted on a computing cluster equipped with the following nodes:

- **C1:** 2 × Intel Xeon E5-2670 v3 (12 cores / 24 threads, Haswell-EP)
- **C2:** 2 × Intel Xeon Gold 5218R (20 cores / 40 threads, Cascade Lake)
- **C3:** 2 × Intel Xeon Gold 6342 (24 cores / 48 threads, Ice Lake-SP)
- **C4:** 2 × Intel Xeon Platinum 8358 (32 cores / 64 threads, Ice Lake-SP)

6.3 Experimental Procedure

To quantify the computational cost of the introduced prediction mechanisms, we defined three encoder variants for comparison:

- **Baseline Version:** An implementation operating exclusively in I-VOP mode (Intra-frame coding), utilizing the same macroblock-based multithreading model but without motion estimation overhead.
- **Hybrid Version:** The proposed full implementation includes Half-Pel motion estimation, P-SKIP mode, and entropy coding.
- **No Half-Pel Version:** An implementation includes motion estimation, P-SKIP mode, and entropy coding, excluding Half-Pel estimation precision.

The execution time measurements cover the entire encoding process, capturing the interval from the start of raw data loading into memory to the completion of the bitstream writing to the disk.

6.4 Performance Analysis

Table 2 presents a detailed comparison of total processing times for the Hybrid (T_H), No-HalfPel (T_N), and Baseline (T_B) variants.

The data reveals a shift in the performance hierarchy compared to simpler parallelization models. The No-HalfPel variant consistently achieves the shortest execution times across all test scenarios. This gain confirms that the computational cost of integer motion estimation is significantly outweighed by the time saved through the aggressive P-SKIP mechanism, which entirely bypasses DCT, quantization, and entropy coding for static blocks.

The Hybrid version demonstrates a processing speed comparable to the Baseline. On platforms C1 and C2, the Hybrid variant consistently outperforms the Baseline, while on the high-performance C3 and C4 nodes, it achieves superiority primarily as the sequence length increases (exceeding 1500 frames). This efficiency is the result of the P-SKIP mechanism, which effectively offsets the additional computational complexity introduced by the Half-Pel interpolation and refinement steps. As shown in the performance analysis, this balanced architecture allows the encoder to achieve high visual fidelity while maintaining high throughput. This balance is further reflected in Table 3, where the No-HalfPel variant appears to achieve higher data reduction than the Hybrid version. This occurs

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

because the Hybrid encoder's higher precision often identifies subtle motion that disqualifies macroblocks from the bit-efficient P-SKIP mode, trading a portion of the compression ratio for the visual fidelity analyzed in the next section.

The results indicate that the advantages of GOP-level parallelism are fully realized as the sequence length grows. At lower frame counts, throughput is constrained by the sequential overhead of task management and bitstream serialization. As more frames are processed, multithreading efficiency increasingly amortizes these fixed costs, resulting in a progressive FPS increase and confirming the system's scalability across multi-core architectures.

Table 2. Computational cost analysis: Total processing time (in seconds) for three encoder variants across four test platforms.

Frames	C1 (12c)			C2 (20c)			C3 (24c)			C4 (32c)		
	T_H	T_N	T_B	T_H	T_N	T_B	T_H	T_N	T_B	T_H	T_N	T_B
250	215.7	69.8	219.1	158.9	50.4	160.3	135.7	44.0	132.5	140.0	45.3	136.4
500	232.3	180.7	223.7	172.9	140.7	173.3	141.3	121.0	133.0	145.6	124.7	136.9
750	256.5	196.1	242.9	192.1	150.7	190.6	144.8	121.7	137.0	146.1	124.7	137.7
1000	292.4	210.9	308.5	203.9	152.7	207.8	145.2	122.2	138.0	146.5	125.2	142.3
1250	335.8	223.8	342.0	226.6	166.4	262.0	147.4	126.5	138.8	148.2	129.8	142.5
1500	430.3	245.4	541.2	254.0	175.6	272.9	165.5	129.1	190.0	150.7	130.1	143.2
1750	536.4	262.1	578.2	284.4	186.4	287.1	194.8	132.4	196.6	155.3	131.1	146.0
2000	576.6	276.6	581.9	296.0	198.3	305.0	210.5	134.7	207.9	176.3	132.7	182.3

T_H: Hybrid (Full), T_N: No-HalfPel, T_B: Baseline (I-VOP)

Table 3. Comparison of compression efficiency for three variants: Hybrid (H), No Half-Pel (N), and Base (B). Reduction columns indicate the percentage gain relative to the Base variant.

Frames	File size [MB]			Data reduction (vs Base)	
	H	N	B	Hybrid	No-HalfPel
250	11.5	5.6	19.0	39.5%	70.5%
500	27.7	17.8	38.8	28.6%	54.1%
750	45.8	34.5	63.9	28.3%	46.0%
1000	55.6	42.5	91.6	39.3%	53.6%
1250	70.6	56.6	115.1	38.7%	50.8%
1500	82.7	62.8	133.1	37.9%	52.8%
1750	95.4	70.4	149.7	36.3%	53.0%
2000	107.6	77.2	207.9	48.2%	62.9%

H: Hybrid (Half-Pel), N: No-HalfPel (Integer), B: Base (I-VOP)

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

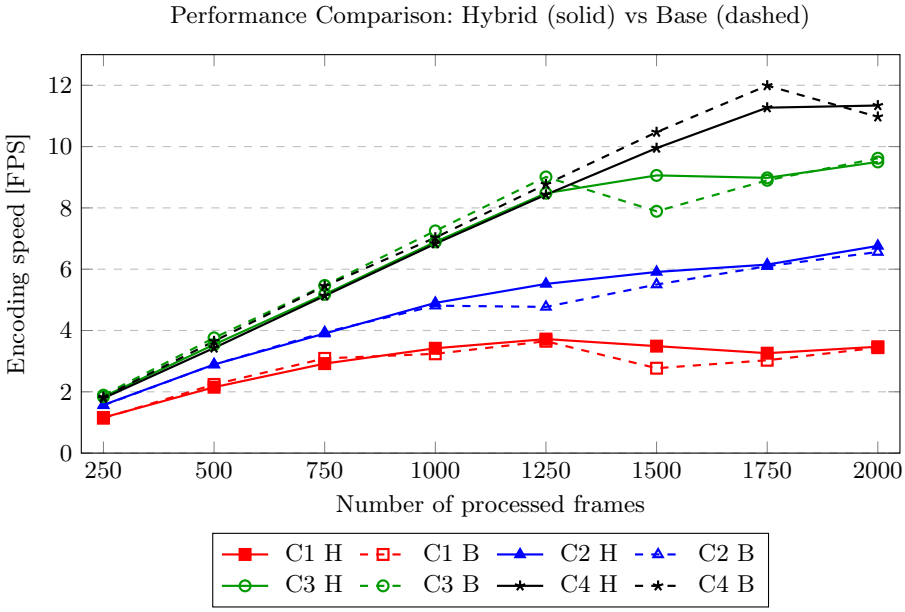

Fig. 2. FPS stability comparison in the GOP-parallel model. The Hybrid variant achieves performance comparable to the Base variant and, in several cases (C2, C4), outperforms it due to the P-SKIP mechanism.

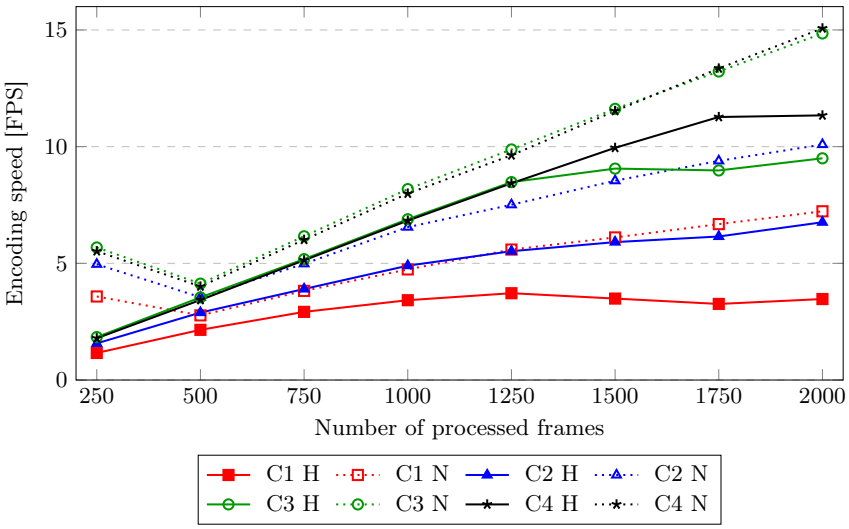

Fig. 3. Impact of removing Half-Pel precision on encoding performance. The No-HalfPel variant achieves significantly higher FPS, at the cost of substantially reduced output quality (see Table 3).

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

6.5 Compression Efficiency Analysis (R-D)

The comparative video quality data presented in Table 4, supported by the R-D curves in Fig. 4 and Fig. 5, clearly illustrate the performance advantage of the Hybrid approach.

The base variant exhibits the lowest coding efficiency. To achieve high reconstruction fidelity at QP=3.0 (PSNR approx. 31.7 dB), it requires a bitrate of 46.11 Mbps. In contrast, the Hybrid variant (blue line) achieves a comparable quality level at 28.00 Mbps, representing a bitrate reduction of approximately 39.3%. At the highest quality setting (QP=1.5), the Hybrid variant maintains an SSIM of 0.9539, while the Baseline requires 66.86 Mbps to provide equivalent perceptual quality. A comparison with the No-HalfPel variant (red line) demonstrates that sub-pixel precision is essential for maintaining structural integrity. Although the No-HalfPel variant produces lower bitrates due to the omission of sub-pixel refinement, it suffers from significant degradation in perceptual quality. For example, at QP=3.0, the SSIM for the No-HP variant decreases to 0.8575, while the Hybrid variant maintains a value above 0.9248. Even at the highest bitrate (QP=1.5), the No-HP version does not achieve an SSIM of 0.91, while the Hybrid version exceeds 0.95. This confirms that Half-Pel interpolation is a fundamental requirement to effectively minimize residual energy and preserve structural details that remain inaccessible to integer-pixel estimation.

Table 4. Comparison of rate-distortion (R-D) characteristics for three encoder variants: Hybrid (H), No Half-Pel (N), and Base (B). Bitrates are normalized to 60 FPS.

QP	Bitrate [Mbps]			PSNR (Y) [dB]			SSIM		
	H	N	B	H	N	B	H	N	B
1.5	36.48	29.86	66.86	32.90	31.26	35.14	0.9539	0.9049	0.9687
3.0	28.00	21.41	46.11	29.72	28.27	31.70	0.9248	0.8575	0.9406
8.0	21.85	17.24	28.37	25.07	23.74	26.27	0.8582	0.7539	0.8537
10.0	21.24	17.12	25.29	24.04	22.78	24.94	0.8387	0.7312	0.8245
15.0	20.61	17.51	22.33	22.22	21.14	22.48	0.8007	0.6886	0.7673
20.0	20.33	17.34	19.44	21.05	20.08	20.79	0.7759	0.6622	0.7243

7 Conclusion

This work presented the design and comprehensive evaluation of a multithreaded hybrid video encoder implemented in Java, focusing on optimizing the trade-off between execution speed and compression efficiency on multi-core architectures. By evolving the architecture from a basic Intra-frame model to a fully featured prediction pipeline utilizing Half-Pel motion estimation, P-SKIP, and entropy coding, we successfully balanced computational complexity with Rate-Distortion

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

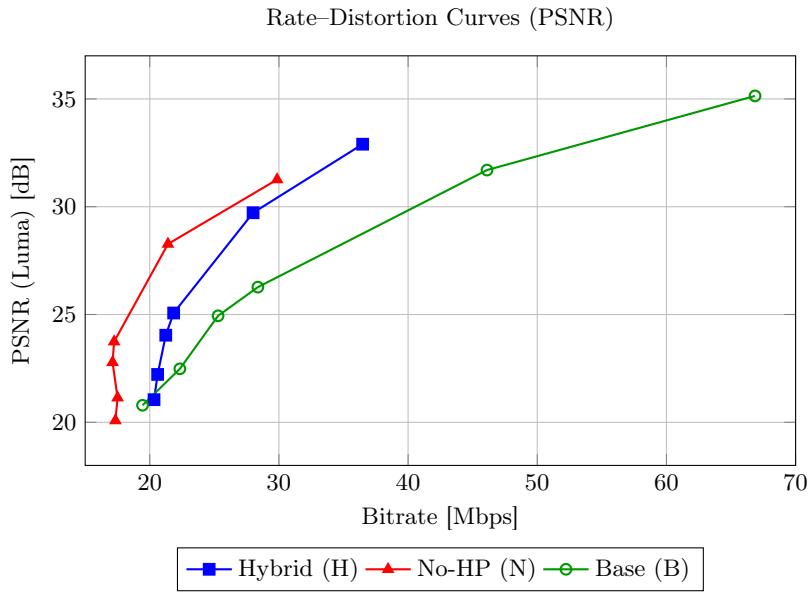

Fig. 4. Rate-distortion comparison. The Hybrid variant (blue) provides the best trade-off, achieving a quality comparable to the Base variant (green) while requiring nearly half the bitrate at key operating points.

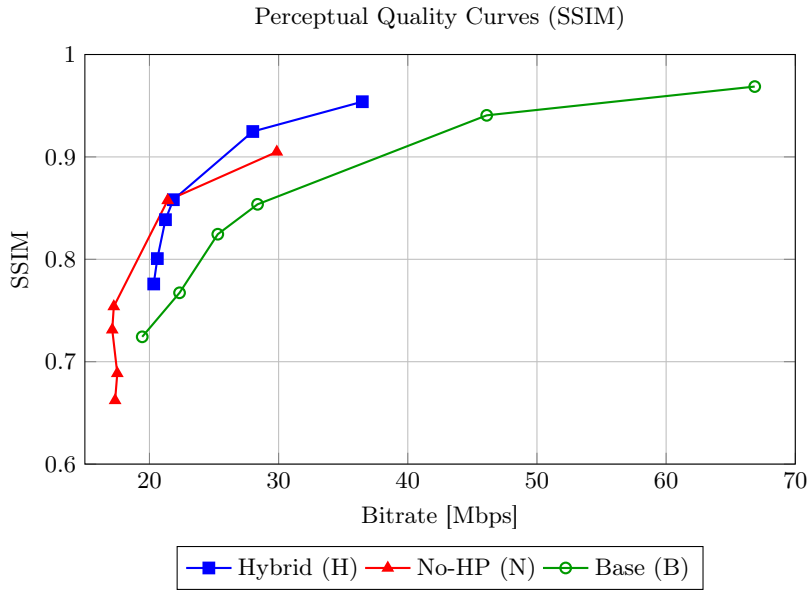

Fig. 5. SSIM analysis. The use of Half-Pel motion estimation allows the Hybrid variant to maintain high structural fidelity while achieving a significant reduction in bitrate.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

performance. The implementation demonstrates that a managed language environment can host sophisticated video processing tasks while maintaining high performance.

Comparative analysis reveals that the computational investment in sub-pixel precision is effectively offset by the architectural optimizations of the hybrid model. Contrary to simpler models, the Hybrid variant achieves a processing speed comparable to, and in several instances exceeding, the Baseline (Intra-only) version. This is primarily due to the P-SKIP mechanism, which bypasses intensive processing for stationary areas, thereby compensating for the overhead of Half-Pel interpolation. This transition yielded substantial bitstream reductions reaching up to 48%. It is important to note that both versions were evaluated using a GOP-parallel multithreading model to ensure architectural parity. This model proved essential for managing temporal dependencies while maintaining high CPU utilization across all evaluated hardware generations.

Quality tests confirm that high quality output is feasible in Java without sacrificing efficiency. The attainment of an SSIM index exceeding 0.95 confirms that the encoder delivers high visual fidelity. The results also highlight that avoiding sub-pixel interpolation is a false optimization; the Integer-precision variant, despite its higher frame rates, suffered from severe perceptual quality degradation.

Furthermore, the experiments indicate that on multi-core systems, the performance gains are increasingly constrained by the sequential phase of the pipeline, particularly bitstream serialization. This behavior aligns with Amdahl's Law [8]: as the parallel motion estimation becomes faster with more cores, the non-parallelizable serialization stage begins to dominate the total execution time.

Future research may focus on extending the model with bi-directional prediction and exploring further optimization through the incubated Java Vector API. Nevertheless, this work offers a grounded perspective on CPU capabilities within a realistic video coding pipeline and serves as a solid foundation for further research into multimedia processing in managed languages.

Acknowledgments. This study was supported by Maria Curie-Skłodowska University.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bylina, B., Okoń, M.: A Multithreaded Java-Based Video Encoder for Multicore Systems. In: Bolanowski, M., Ganzha, M., Maciaszek, L., Paprzycki, M., Ślęzak, D. (eds.) Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS). ACSIS, vol. 43, pp. 659–664 (2025). https://doi.org/10.15439/2025F1886
2. Oracle Corporation: Java Platform, Standard Edition 8 API Specification - Class BufferedImage. `https://docs.oracle.com/javase/8/docs/api/java/awt/image/BufferedImage.html` (2024). Accessed: 2026-02-18

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23

3. Moreira, J.E., Midkiff, S.P., Gupta, M., Artigas, P.V., Snir, M., Lawrence, R.D.: Java programming for high-performance numerical computing. IBM Systems Journal 39(1), 21–56 (2000). https://doi.org/10.1147/sj.391.0021
4. Sankaraiah, S., Shuan, L.H., Eswaran, C., Abdullah, J.: Performance Optimization of Video Coding Process on Multi-Core Platform Using GOP Level Parallelism. International Journal of Parallel Programming 42(6), 931–947 (2014). https://doi.org/10.1007/s10766-013-0288-5
5. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4), 600–612 (2004). https://doi.org/10.1109/TIP.2003.819861
6. Huffman, D.A.: A Method for the Construction of Minimum-Redundancy Codes. Proceedings of the IRE 40(9), 1098–1101 (1952). https://doi.org/10.1109/JRPROC.1952.273898
7. Chee, C.S., Jambek, A.B., Hussin, R.: Review of energy efficient block-matching motion estimation algorithms for wireless video sensor networks. In: Proceedings of the 2012 IEEE Symposium on Computers & Informatics (ISCI), pp. 241–246 (2012). https://doi.org/10.1109/ISCI.2012.6222702
8. Amdahl, G.M.: Validity of the single processor approach to achieving large scale computing capabilities. In: AFIPS '67 (Spring) Proceedings of the April 18-20, 1967, spring joint computer conference, pp. 483–485. ACM (1967). https://doi.org/10.1145/1465482.1465560

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29909-3_23