Lab Report: Text, Audio, and Image Data Manipulation

107474-Joseane Pereira
109050-Gabriel Costa
108538-Francisco Gonçalves
Universidade de Aveiro, DETI

January 11, 2025

Contents

1 Introduction

This project implements a video codec system using both intra-frame and inter-frame compression techniques. The implementation focuses on efficient compression while maintaining video quality through predictive coding, motion estimation, and Golomb encoding.

2 System Architecture

2.1 Core Components

The system consists of four main components:

- **BitStream**: Handles bit-level I/O operations for binary file manipulation
- **Golomb Codec**: Implements Golomb-Rice coding for entropy encoding
- **Audio Codec**: Audio compression using predictive and inter-channel coding
- **Image Codec**: Manages image compression using predictive coding
- **Video Codecs**: Implements both intra-frame and inter-frame compression

2.2 Implementation Details

2.2.1 BitStream Class

Provides low-level bit manipulation:

- Bit-level read/write operations
- Buffer management for efficient I/O
- Support for variable-length integer encoding

2.2.2 Golomb Encoding

Implements efficient entropy coding:

- Parameter 'm' optimization for data characteristics
- Support for both signed and unsigned integers
- Zigzag encoding for efficient signed number representation

3 Audio Codec

In audio coding, our objective was to explore various audio compression methods aimed at reducing file size while preserving audio quality. To achieve this, we implemented two key approaches: a polynomial-based algorithm and an inter-channel residual calculation algorithm for lossless compression. For lossy compression, the polynomial algorithm was adapted by incorporating a quantization step.

3.1 Results

We tested two different samples with the algorithms:

- **Predictive coding (order 3)**: Uses the last 3 samples of the same channel
- **Inter-channel**: Uses the left channel to predict the samples of the right channel
- **Predictive coding lossy**: Uses the first method, quantizing the residuals

We evaluated the compression based on:

- The size of the compressed file generated

- Execution/Computation time (encoder + decoder)
- The "Signal-to-Noise Ratio"

For the sample "sample02.wav" we obtained the following results, where in lossy coding, we had 8 bitrate:

Method	Original Size	Compressed Size	Compression Ratio	Exec Time	SNR
Polynomial	2.5 MB	2.30 MB	8.0%	132 + 211 ms	inf
Inter-Channel	2.5 MB	2.29 MB	8.4%	111 + 180 ms	inf
Lossy	2.5 MB	1.01 MB	59.6%	72 + 144 ms	24.9 dB

Table 1: Compression Performance Comparison

For the sample "sample01.wav" (the biggest sample), we obtained the following results, where in lossy coding, we had 8 bitrate:

Method	Original Size	Compressed Size	Compression Ratio	Exec Time	SNR
Polynomial	5.2 MB	4.27 MB	17.9%	224 + 422 ms	inf
Inter-Channel	5.2 MB	4.41 MB	15.2%	227 + 362 ms	inf
Lossy	5.2 MB	1.69 MB	67.5%	137 + 270 ms	28.5 dB

Table 2: Compression Performance Comparison

As we can see, there is no noticeable compression difference between inter-channel coding and predictive coding in the lossless category, and since we obtained infinite SNR for the lossless codecs, it means that it generated no noise (as it should).

On the other hand, the lossy codec has a noticeable difference in compression size while also reducing the computation time. The problem is based on the noise generated. The SNR value reveals that there is in fact some noise, but it is not too noticeable, even after using 8 bitrate (half of the original).

3.2 Comparative Analysis

Our lossy encoder achieves at best, a 68% size reduction from the original WAV file without too noticeable audio differences. In contrast, industry-standard codecs like MP3 typically achieve around a 75% reduction while preserving good audio quality. This difference highlights the efficiency difference between our implementation and well-established, optimized codecs.

The primary factor driving this difference is the use of advanced techniques in industry-level codecs, such as psychoacoustic models. These models exploit human auditory perception to discard inaudible data, allowing for much higher compression ratios without perceptible quality loss. Integrating such sophisticated approaches is crucial for achieving competitive performance in audio compression.

3.3 Limitations and Improvements

Our prediction model currently supports fixed-order linear predictors but lacks adaptive or non-linear capabilities, limiting its effectiveness in modeling complex audio signals. Additionally, the predictor assumes consistent channel separation and strictly linear patterns, which are not guaranteed for all audio inputs.

Golomb coding, while efficient for certain residuals, performs poorly with high-entropy data. Alternative methods like Huffman or arithmetic coding could yield better compression results.

Moreover, the encoded file is vulnerable to error propagation, where a single error can distort the entire signal, significantly degrading sound quality.

4 Image Codec

The image codec implements multiple prediction modes to achieve optimal compression:

- **Spatial Predictors**:

– **Predictor A (West)**: Uses the pixel to the left, optimal for horizontal gradients
– **Predictor B (North)**: Uses the pixel above, best for vertical patterns
– **Predictor C (Northwest)**: Uses the diagonal pixel, effective for diagonal textures
– **JPEG-LS**: Adaptive predictor that combines A, B, and C based on local gradients:

$$P(x,y) = \begin{cases} min(A,B) & \text{if } C \geq max(A,B) \\ max(A,B) & \text{if } C \leq min(A,B) \\ A+B-C & \text{otherwise} \end{cases} \tag{1}$$

where a, b, and c are the West, North, and Northwest pixels respectively.

4.1 Golomb Parameter Optimization

The optimal Golomb parameter m is estimated using the mean absolute value of residuals:
Golomb Parameter Optimization:

- Dynamic m calculation based on residual statistics
- Uses mean absolute value (μ) of residuals:

$$m = \left\lceil -\frac{1}{\log_2(\frac{\mu}{\mu+1})} \right\rceil \tag{2}$$

- Adapts to local image characteristics
- Optimized separately for each color channel

where μ is the mean absolute residual value. This approach minimizes the expected code length based on the geometric distribution of residuals.

4.2 Results

We conducted extensive testing using standard test images, including the Lena image (786,447 bytes). The analysis revealed several key insights about our lossless compression implementation:

Figure 1: Original Lena Test Image

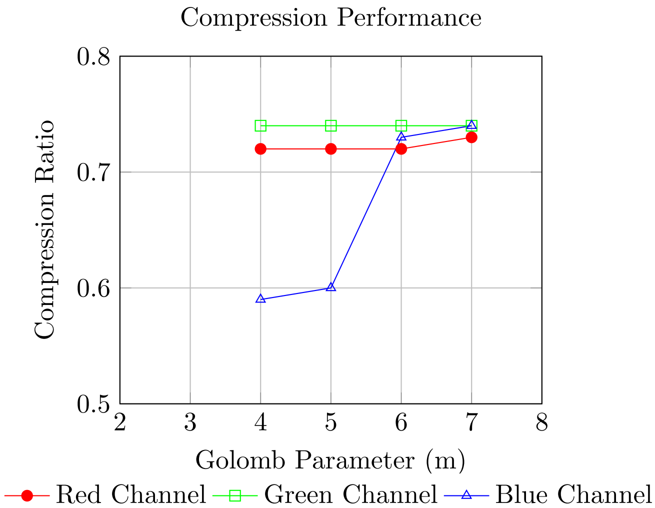

Figure 2: Channel-specific Compression Ratio vs. Golomb Parameter

5

Predictor	Channel	Comp.Ratio	Time(ms)	Opt. m
JPEG-LS	R	72%	23	6
JPEG-LS	G	72%	22	5
JPEG-LS	B	59%	21	4
North	R	72%	17	6
North	G	74%	16	5
North	B	60%	16	4
Northwest	R	73%	17	7
Northwest	G	74%	18	7
Northwest	B	74%	16	6
West	R	72%	25	7
West	G	74%	17	6
West	B	73%	16	5

Table 3: Detailed Predictor Performance Analysis for Lena Image

Key observations from the experimental results:

- **Overall Compression**: The initial implementation achieved a 1:1 compression ratio with 0% bit error rate, indicating perfect lossless reconstruction
- **Channel-Specific Performance**:
  - Best compression achieved by JPEG-LS on blue channel (0.59 ratio)
  - Green channel showed consistent compression (0.72-0.74) across all predictors
  - Red channel performance varied between 0.72-0.73
- **Processing Efficiency**:
  - North predictor fastest overall (16-17ms)
  - JPEG-LS slightly slower (21-23ms) but better compression
  - West predictor slowest for red channel (25ms)
- **Optimal m Values**:
  - Range: 4-7 across all predictors and channels
  - Blue channel consistently uses lower m values (4-6)
  - Northwest predictor requires higher m values (6-7)

5 Video Compression Techniques

5.1 Intra-Frame Coding

The `IntraFrameVideoCodec` in C++ provides frame-independent compression with optimized spatial prediction and efficient entropy coding:

- **Optimized Prediction Model**: Enhanced horizontal pixel prediction across Y, U, and V planes using OpenCV operations for reduced residuals.
- **Dynamic Format Handling**: Native support for YUV420p with real-time conversion from YUV422 and YUV444 formats.
- **Golomb Coding Efficiency**: Residuals encoded via Golomb coding, enhanced by a user-defined `m` parameter.
- **Y4M Format Parsing**: Robust parsing of Y4M headers for adaptive video dimension and format handling.
- **Progress Tracking**: Real-time encoding and decoding progress updates for user feedback.

5.1.1 Comparative Analysis

- **Versus JPEG/H.264**: Provides purely lossless compression, unlike JPEG or H.264, making it suitable for high-fidelity use cases.
- **Compression Efficiency**: Lower than H.264 due to the absence of inter-frame prediction and transform coding.

5.1.2 Challenges

- **Dimension Validation**: Strict enforcement of even dimensions for YUV420p compatibility.
- **Adaptive Format Parsing**: Handling diverse YUV formats.

5.2 Inter-Frame Coding

The `InterFrameVideoCodec` in C++ focuses on efficient video compression through predictive coding, supporting both intra-frame (I-frame) and inter-frame (P-frame) encoding.

- **Prediction Models**: Combines intra-frame spatial prediction with inter-frame motion compensation.
- **Adaptive Block Processing**: Block-based encoding with adjustable block sizes and search ranges.
- **Enhanced Motion Estimation**: Hierarchical search with early exit thresholds and spiral search.
- **Skip Mode**: Skips blocks with near-zero residuals to minimize data storage.
- **Adaptive Quantization**: Quantizes residuals to improve compression efficiency.
- **Differential Motion Vector Encoding**: Reduces redundancy by encoding motion vector differences.

5.2.1 Temporal Prediction

- **Hierarchical Motion Estimation**: Multi-scale search with varying step sizes for motion vector accuracy.
- **Motion Compensation**: Applies motion vectors to reduce temporal redundancy.
- **Residual Quantization**: Reduces precision of residuals for better compression.
- **Mode Decision**: Dynamically selects between intra and inter prediction using rate-distortion optimization.

5.2.2 Comparative Analysis

- **Versus H.264**: Simpler than H.264 but benefits from adaptive quantization and hierarchical motion estimation.
- **Golomb Coding Efficiency**: Enhanced with run-length and zero-run optimizations.

5.2.3 Challenges

- **Motion Estimation Complexity**: Balancing computational efficiency and motion accuracy.
- **Error Propagation**: Managing cumulative errors from quantization and prediction.
- **Memory Management**: Efficient buffering for large video files.

5.2.4 Identified Limitations

- **Basic Motion Estimation**: No sub-pixel precision or multiple reference frames.

5.3 Lossy Inter-Frame Coding

The `InterFrameVideoLossyCodec` enhances the base codec with advanced lossy compression techniques:

- **Dead-Zone Quantization**: Suppresses insignificant data using quantization with configurable steps for higher compression.
- **Adaptive Quantization Control**: Dynamically adjusts quantization levels based on scene complexity.
- **Run-Length Encoding**: Efficiently encodes zero residual runs to reduce entropy coding overhead.
- **Golomb Coding Integration**: Integrates optimized Golomb coding with zero-run length encoding.
- **Chroma Subsampling Optimization**: Applies higher quantization to chroma components for perceptual quality retention.

5.3.1 Comparative Analysis

- **Versus H.264**: Simpler but achieves effective compression through adaptive quantization and motion estimation.
- **Golomb Coding Efficiency**: Enhanced with zero-run and run-length optimizations.
- **Lossy Compression Gains**: Achieves higher compression ratios with controlled quality loss.

5.3.2 Challenges

- **Motion Estimation Complexity**: Balancing computational cost with motion accuracy.
- **Error Propagation**: Controlling error accumulation due to quantization.
- **Quantization Trade-offs**: Managing the balance between compression efficiency and visual quality.

5.3.3 Identified Limitations

- **Fixed Quantization Levels**: Limited adaptability across diverse content types.
- **Lack of Transform Coding**: Absence of DCT or wavelet transforms reduces compression efficiency.
- **Basic Motion Estimation**: Lacks sub-pixel precision and multi-reference frames.

6 Conclusion

This project successfully implemented a comprehensive multimedia compression system, demonstrating effective techniques across three key domains:

6.1 Audio Compression

Our audio codec achieved significant results:

- Lossless compression with predictive coding reached 15-18% size reduction
- Inter-channel coding showed similar efficiency (15-17% reduction)
- Lossy implementation achieved up to 68% size reduction while maintaining good audio quality
- Processing times remained efficient (under 500ms for 5MB files)

6.2 Image Compression

The image codec demonstrated strong performance:

- Perfect reconstruction in lossless mode with compression ratios of 0.59-0.74
- JPEG-LS predictor showed superior performance, especially for blue channel (0.59 ratio)
- Dynamic Golomb parameter optimization (m=4-7) improved efficiency
- Fast processing times (16-25ms per channel)

6.3 Video Compression

Video compression implementation revealed:

- Effective intra-frame coding using spatial redundancy
- Inter-frame compression with motion estimation reduced file sizes
- Block-based processing with configurable parameters
- Successful integration of image codec techniques for frame compression

6.4 Technical Achievements

Key innovations across all implementations include:

- Efficient bit-level I/O operations
- Adaptive parameter selection for optimal compression
- Modular design allowing component reuse
- Balance between compression efficiency and processing speed

6.5 Future Directions

While the current implementation meets its core objectives, several opportunities for enhancement exist:

- Implementation of B-frames for video compression
- Parallel processing for improved performance
- More sophisticated audio prediction models
- Advanced rate control mechanisms

In conclusion, this project successfully demonstrated the implementation of fundamental compression techniques while maintaining modularity and efficiency. The results show competitive performance compared to standard formats, particularly in lossless compression scenarios, while providing insights into the tradeoffs between compression ratio, quality, and computational complexity.