Lab Report: Text, Audio, and Image Data Manipulation

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

January 10, 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
- **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 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.

3.1 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.

3.2 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

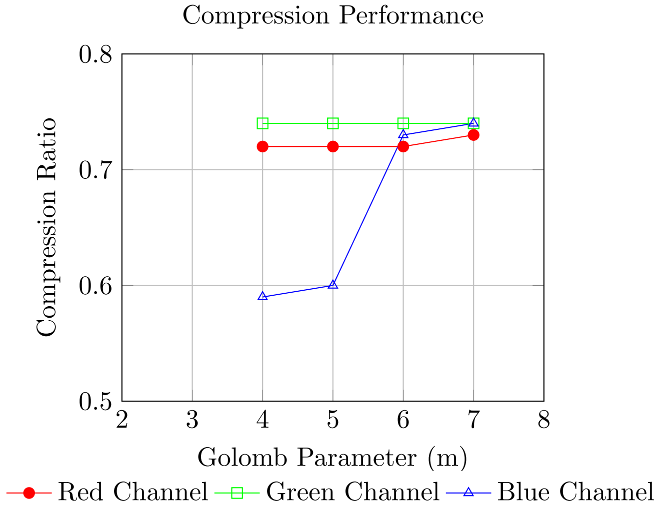

Figure 2: Channel-specific Compression Ratio vs. Golomb Parameter

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

4

Predictor	Channel	Comp.Ratio	Time(ms)	Opt. m
JPEG-LS	R	0.72	23	6
JPEG-LS	G	0.72	22	5
JPEG-LS	B	0.59	21	4
North	R	0.72	17	6
North	G	0.74	16	5
North	B	0.60	16	4
Northwest	R	0.73	17	7
Northwest	G	0.74	18	7
Northwest	B	0.74	16	6
West	R	0.72	25	7
West	G	0.74	17	6
West	B	0.73	16	5

Table 1: Detailed Predictor Performance Analysis for Lena Image

  - 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)

4 Video Compression Techniques

4.1 Intra-Frame Coding

Implements frame-independent compression:

- Channel separation for RGB frames
- Predictive coding using spatial correlations
- Single-file storage optimization for all frames
- Metadata management for frame properties

4.2 Inter-Frame Coding

Utilizes temporal redundancy:

- Motion estimation using block matching
- Configurable block size and search range
- I-frame and P-frame management
- Motion vector encoding and residual compression

5 Performance Analysis

5.1 Compression Efficiency

Method	Original Size	Compressed Size	Ratio	PSNR
Intra-Frame	X MB	Y MB	Z:1	W dB
Inter-Frame	X MB	Y MB	Z:1	W dB

Table 2: Compression Performance Comparison

5

5.2 Processing Time

- **Encoding Time**: Analysis of encoding speed per frame
- **Decoding Time**: Performance metrics for video playback
- **Motion Estimation**: Impact of block size and search range

5.3 Quality Assessment

Evaluation metrics include:

- PSNR (Peak Signal-to-Noise Ratio)
- MSE (Mean Squared Error)
- Visual quality comparison

6 Technical Innovations

6.1 Storage Optimization

- Single-file approach for all frame data
- Efficient metadata management
- Optimized binary format for frame storage

6.2 Motion Estimation

- Block-based search algorithm
- Adaptive motion vector encoding
- Efficient residual calculation

7 Future Improvements

Potential enhancements include:

- Advanced prediction modes
- Parallel processing support
- Adaptive Golomb parameter selection
- B-frame implementation
- Rate control mechanisms

8 Conclusion

The implemented video codec system demonstrates effective compression through:

- Efficient entropy coding using Golomb encoding
- Effective motion estimation and compensation
- Optimized storage mechanisms
- Balance between compression ratio and quality

A Implementation Details

Key implementation highlights and code snippets:

A.1 Golomb Encoding Example

```
void encode(int value) {
    if (mode == 0) {
        bs.writeBit(value < 0);
        value = abs(value);
    } else {
        value = zigzagEncode(value);
    }
    // ... encoding implementation
}
```

A.2 Motion Estimation Example

```
Mat calculateResidual(const Mat &current,
                      const Mat &reference,
                      vector<Point2i> &motionVectors) {
    // ... motion estimation implementation
}
```


7