Project 8 Roadmap

Color Image Predictive Coding with Feedback Loop

IHT3 - 2D and 3D Visual Data Compression

Contents

1 Introduction and Objectives

Project 8 involves implementing a predictive coding system for RGB color images using causal windows and a feedback loop. The main objective is to reduce signal entropy by exploiting spatial and inter-color correlations.

1.1 Fundamental Principles

- **Inter-plane prediction**: Exploiting correlations between R, G, B channels
- **Causal window**: Using only already processed pixels (raster-scan order)
- **Feedback loop**: Prediction based on reconstructed values
- **AR modeling**: Optimal coefficient calculation using least squares

1.2 Prediction Equations

$$\hat{R}(i,j) = \sum_{k=1}^{9} r_k \cdot \text{neighbors}_k \tag{1}$$

$$\hat{G}(i,j) = \sum_{k=1}^{10} g_k \cdot \text{neighbors}_k \tag{2}$$

$$\hat{B}(i,j) = \sum_{k=1}^{11} b_k \cdot \text{neighbors}_k \tag{3}$$

2 Phase 1: Preparation and Base Structure

Phase 1: Analysis and Architecture

2.1 Analysis and Understanding

- ☐ Study prediction windows for each plane (R, G, B)
- ☐ Understand prediction equations (1), (2), (3)
- ☐ Analyze provided function prototypes
- ☐ Identify inter-plane dependencies

2.2 Code Structure

- ☐ Create file structure (headers, sources)
- ☐ Define data structures for:
  - RGB images (2D matrices)
  - Prediction coefficients
  - Prediction errors
  - Covariance matrices
- ☐ Implement basic utilities (image read/write)
- ☐ Prepare memory management functions

3 Phase 2: Causal Windows Implementation

Phase 2: Prediction Windows

3.1 Causal Neighbor Extraction

- ☐ Implement extraction for R plane: 9 neighbors
  - 3 neighbors from R plane: $R(i-1,j)$, $R(i,j-1)$, $R(i-1,j-1)$
  - 3 neighbors from G plane: $G(i-1,j)$, $G(i,j-1)$, $G(i-1,j-1)$
  - 3 neighbors from B plane: $B(i-1,j)$, $B(i,j-1)$, $B(i-1,j-1)$
- ☐ Implement extraction for G plane: 10 neighbors (9 + 1 from R)
- ☐ Implement extraction for B plane: 11 neighbors (9 + 2 from R and G)

3.2 Border Handling

- ☐ Handle border pixels (zero padding or special conditions)
- ☐ Implement correct raster-scan traversal
- ☐ Handle edge cases (top-left corner, first row, first column)

WARNING: Causal Window Attention

Strictly respect raster-scan order. Never use "future" pixels that have not yet been processed in the traversal order.

4 Phase 3: Optimal Coefficient Calculation

Phase 3: AR Modeling and Optimization

4.1 AR Modeling

- ☐ Implement autocorrelation matrix calculation
- ☐ For each predictor, build the linear equation system:

$$\mathbf{R} \cdot \mathbf{r} = \mathbf{p}_R \tag{4}$$
$$\mathbf{R} \cdot \mathbf{g} = \mathbf{p}_G \tag{5}$$
$$\mathbf{R} \cdot \mathbf{b} = \mathbf{p}_B \tag{6}$$

where $\mathbf{R}$ is the autocorrelation matrix and $\mathbf{p}$ are the cross-correlation vectors

4.2 System Resolution

- ☐ Implement resolution using LU or Cholesky decomposition
- ☐ Calculate optimal coefficients for each predictor
- ☐ Handle singular or ill-conditioned matrices
- ☐ Validate numerical stability of solutions

5 Phase 4: Prediction Strategies

Phase 4: Global and Local Approaches

5.1 Global Strategy

- ☐ Calculate a unique set of coefficients for the entire image
- ☐ Implement global `predictionRGB()` function
- ☐ Optimize for reduced computational complexity

5.2 Local Strategy

- ☐ Partition image into 32×32 pixel blocks
- ☐ Calculate specific coefficients for each block
- ☐ Handle border blocks (non-standard size)
- ☐ Implement transition management between blocks

6 Phase 5: Encoding and Decoding

Phase 5: Main Algorithm Implementation

6.1 Encoding Procedure

- ☐ Implement `predictionRGB()`:
  - Calculate means (R_{med}, G_{med}, B_{med})
  - Center signals (zero mean)
  - Calculate prediction errors
  - Uniform quantization with parameter δ
- ☐ Maintain feedback loop (reconstructed values)

6.2 Decoding Procedure

- ☐ Implement `predictionRGB_inv()`:
  - Reconstruction from quantized errors
  - Add means to obtain final signals
  - Respect feedback loop
- ☐ Validate perfect reconstruction ($\delta = 0$)

WARNING: Critical Feedback Loop

CRUCIAL: Use reconstructed values for prediction, never originals. Maintain absolute consistency between encoder and decoder.

7 Phase 6: Evaluations and Comparisons

Phase 6: Performance Analysis

7.1 Performance Metrics

- ☐ Calculate entropy of prediction errors
- ☐ Measure entropy reduction compared to original signal
- ☐ Evaluate visual quality (MSE, PSNR)
- ☐ Analyze error distribution

7.2 Required Comparisons

- ☐ **Global vs local strategy**: Compare entropy reduction
- ☐ **Window size effect**: Test with reduced windows
- ☐ **Cross-prediction vs band-by-band**:
  - Implement prediction using only current plane
  - Compare performance of both approaches

7.3 Experimental Validation

- ☐ Test on multiple reference images
- ☐ Verify perfect reconstruction ($\delta = 0$)
- ☐ Analyze quantization step δ impact
- ☐ Document results with graphs and tables

8 Phase 7: Optimizations and Finalization

Phase 7: Refinement and Documentation

8.1 Optimizations

- ☐ Optimize matrix calculations (BLAS/LAPACK if available)
- ☐ Consider parallelization (independent blocks)
- ☐ Implement efficient memory management
- ☐ Profile and optimize bottlenecks

8.2 Testing and Documentation

- ☐ Develop unit tests for each component
- ☐ Create comprehensive technical documentation
- ☐ Write results analysis report
- ☐ Prepare visual demonstrations

9 Critical Attention Points

WARNING: Feedback Loop

CRUCIAL: Use reconstructed values for prediction, not originals. Maintain absolute encoder/decoder consistency.

WARNING: Causal Windows

Strictly respect raster-scan order. Never use "future" pixels.

WARNING: Mean Management

Center signals before prediction. Transmit means to decoder for correct reconstruction.

INFO: Numerical Stability

Handle ill-conditioned matrices. Use appropriate data types (double precision). Verify convergence of iterative algorithms.

10 Expected Deliverables

1. **Complete source code** with all required functions
2. **Experimental results**:
   - Global/local strategy comparison
   - Window size impact
   - Cross-prediction vs band-by-band

3. **Performance analysis** (entropy, visual quality)
4. **Technical documentation** and detailed report

11 Functions to Implement

11.1 Main Prototypes

```
int predictionRGB(double **r, double **g, double **b,
                  double **err_r, double **err_g, double **err_b,
                  int Width, int Height, double delta,
                  double &Rmed, double &Gmed, double &Bmed);

int predictionRGB_inv(double **err_r, double **err_g, double **err_b,
                      double **r_rec, double **g_rec, double **b_rec,
                      int Width, int Height,
                      double Rmed, double Gmed, double Bmed);
```

11.2 Auxiliary Functions

- Optimal prediction coefficient calculation
- Causal window extraction
- Covariance matrix management
- Quantization/dequantization
- Entropy calculation