Ultraspectral Data Compression

A Dissertation Presented
by
Rolando Herrero
to
The Department of Electrical and Computer Engineering

in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in the field of
Electrical Engineering

Northeastern University
Boston, Massachusetts

October, 2012
Abstract

Ultraspectral images capture 2D data tuned at different wavelengths across the mid infrared electromagnetic spectrum. Atmospheric Infrared Sounder (AIRS) remote sensors are ultraspectral sensors that generate images with thousands of highly correlated bands and are considered the future of spectroscopy. The major application of AIRS is the acquisition of atmospheric parameters such as gases, temperature and other quantities to perform climate and weather forecast. Because typical AIRS images are well over 40 MB in size and because they are captured in remote locations data compression (before transmission) becomes a very critical issue. Existent compression studies of AIRS data adapt generic multispectral image compression techniques (not necessarily ultraspectral) but do not take into account the particular nature of ultraspectral images. Most of them do not consider correlation beyond one band, use fixed linear prediction (leading to significant distortion) and are not optimized to overcome time and space complexity constraints. Moreover compression studies do not provide analytical models of rate-distortion either.

This dissertation focuses on presenting a whole new architecture for the compression of ultraspectral data presenting sound mathematical models that can be used to describe a set of algorithms and their practical implementation. Specifically we will (1) present a new preprocessing reversible stage that will rearrange the data to make it more efficient when the compression stage is performed; (2) present a new linear prediction based compression stage that will improve the compression rate of any given distortion when compared to literature ultraspectral data compression techniques. This involves defining a distortion measure and its effect on real applications; (3) define a mathematical model to approximate the rate-distortion of the compression stage and compare it against the real performance of the proposed algorithm; (4) compare the performance of the overall architecture against that of other state of the art compression schemes.

To summarize it can be said that the results of this dissertation will contribute to the understanding of ultraspectral compression as well as the introduction of a novel ultraspectral codec.
## Contents

1 Introduction .................................................................................................................. 18
   1.1 Multispectral Imaging ......................................................................................... 18
1.2 Multispectral Data ................................................................................................. 23
   1.2.1 AVIRIS ........................................................................................................ 23
   1.2.2 AIRS .......................................................................................................... 24
   1.2.3 AIRS Processing .......................................................................................... 25
1.3 Ultraspectral Data Compression .......................................................................... 26
   1.3.1 Lossless vs Lossy Compression .................................................................... 26
   1.3.2 Distortion Measure ...................................................................................... 27
   1.3.3 Architecture for Ultraspectral Data Compression ......................................... 30
1.4 Objectives and Contribution .............................................................................. 30

2 Review of Literature .................................................................................................... 32
   2.1 Basic Techniques ............................................................................................... 32
      2.1.1 Entropy Coding ......................................................................................... 32
      2.1.2 Vector Quantization .................................................................................. 33
      2.1.3 Transform Coding ...................................................................................... 34
         2.1.3.1 PCA ................................................................................................... 34
         2.1.3.2 DCT ................................................................................................... 34
         2.1.3.3 DWT ................................................................................................... 36
      2.1.4 Linear Mixture Analysis .......................................................................... 36
      2.1.5 Linear Prediction ....................................................................................... 36
   2.2 Preprocessing Techniques .................................................................................. 44
      2.2.1 Band Ordering ............................................................................................ 44
      2.2.2 Bias Adjusted Reordering ......................................................................... 45
   2.3 Compression Techniques ................................................................................... 45
      2.3.1 Entropy Coding ......................................................................................... 45
      2.3.2 Vector Quantization .................................................................................. 46
      2.3.3 Transform Coding ...................................................................................... 47
         2.3.3.1 PCA ................................................................................................... 47
List of Figures

1.1 Hyperspectral Data Structure ........................................ 19
1.2 Reflectance Curve .................................................. 19
1.3 Hyperspectral Pixel Vector .......................................... 20
1.4 Spectral Solar Irradiance ............................................. 21
1.5 Atmospheric Absorption ............................................. 22
1.6 AVIRIS .............................................................. 24
1.7 AIRS ............................................................... 25
1.8 Encoder .............................................................. 29
1.9 Decoder .............................................................. 29

2.1 PCA axis rotation: (a) before, (b) after ............................ 35
2.2 DWT ............................................................... 35
2.3 LP ................................................................. 36
2.4 System Identification ................................................ 37
2.5 Barnwell window ..................................................... 40
2.6 AR encoder .......................................................... 40
2.7 AR decoder .......................................................... 41
2.8 MA encoder .......................................................... 41
2.9 MA decoder .......................................................... 41
2.10 AR encoder ......................................................... 42
2.11 AR decoder .......................................................... 42
2.12 MA encoder .......................................................... 43
2.13 MA decoder .......................................................... 43
2.14 FB encoding ........................................................ 48
2.15 3D ADPCM ........................................................ 49
2.16 JPEG-7 ............................................................. 50
2.17 JPEG-LS ............................................................ 50
2.18 SLSQ Contexts ....................................................... 52

3.1 March 3, 2004 Granule 16 ............................................. 56
3.2 March 3, 2004 Granule 82 .............................................. 56
3.3 March 3, 2004 Granule 126 ............................................. 57
3.4 March 3, 2004 Granule 151 ............................................. 57
3.5 March 3, 2004 Granule 193 ............................................. 58
3.6 Voiced frame ............................................................. 60
3.7 Voiced frame (zoom) ..................................................... 60
3.8 Two-Level Signal Decomposition/Reconstruction .......... 63
3.9 Row-Column DWT Computation .................................. 63
3.10 Second Level Decomposition ..................................... 63
3.11 Subband Decomposition: March 3, 2004 Granule 16 .......... 64
3.12 Subband Decomposition: March 3, 2004 Granule 82 .......... 65
3.13 Subband Decomposition: March 3, 2004 Granule 126 ...... 65
3.15 Subband Decomposition: March 3, 2004 Granule 151 ...... 66
3.16 Subband Decomposition: March 3, 2004 Granule 16 (Histogram Equalization) 67
3.17 Subband Decomposition: March 3, 2004 Granule 82 (Histogram Equalization) 67
3.18 Subband Decomposition: March 3, 2004 Granule 126 (Histogram Equalization) 68
3.19 Subband Decomposition: March 3, 2004 Granule 151 (Histogram Equalization) 68
3.20 Subband Decomposition: March 3, 2004 Granule 193 (Histogram Equalization) 69
3.21 Block layout ............................................................ 72
3.22 Distribution of Bias Adjustment (blocks size 2 × 2) .......... 73
3.23 Distribution of Bias Adjustment (blocks size 5 × 5) .......... 73
3.24 Distribution of Bias Adjustment (blocks size 10 × 10) .... 74
3.25 Bias Adjustment at Block (0, 0) .................................. 74
3.26 Bias Adjustment at Band 0 ......................................... 75
3.27 Band Range Distribution .......................................... 77
3.28 Band mapping ......................................................... 78
3.29 Zig-Zag Scanning (type 1) .......................................... 78
3.30 Zig-Zag Scanning (type 2) .......................................... 78
3.31 Zig-Zag Scanning (type 3) .......................................... 78
3.32 Zig-Zag Scanning (type 4) .......................................... 79
3.33 Progressive Scanning (type 1) ..................................... 79
3.34 Progressive Scanning (type 2) ..................................... 79
3.35 Linear Scanning (type 1) ........................................... 79
3.36 Linear Scanning (type 2) ........................................... 80
3.37 Unidimensional AIRS Data (2 × 2 blocks) .................... 82
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.38</td>
<td>Unidimensional AIRS Data (zoom) (2 × 2 blocks)</td>
<td>83</td>
</tr>
<tr>
<td>3.39</td>
<td>Unidimensional AIRS Data (5 × 5 blocks)</td>
<td>83</td>
</tr>
<tr>
<td>3.40</td>
<td>Unidimensional AIRS Data (zoom) (5 × 5 blocks)</td>
<td>84</td>
</tr>
<tr>
<td>3.41</td>
<td>Unidimensional AIRS Data (10 × 10 blocks)</td>
<td>84</td>
</tr>
<tr>
<td>3.42</td>
<td>Unidimensional AIRS Data (zoom) (10 × 10 blocks)</td>
<td>85</td>
</tr>
<tr>
<td>3.43</td>
<td>Autocorrelation Output (2 × 2 blocks)</td>
<td>87</td>
</tr>
<tr>
<td>3.44</td>
<td>Autocorrelation Output (5 × 5 blocks)</td>
<td>88</td>
</tr>
<tr>
<td>3.45</td>
<td>Autocorrelation Output (10 × 10 blocks)</td>
<td>88</td>
</tr>
<tr>
<td>3.46</td>
<td>Real Data vs Laplacian λ = 150 (2 × 2 blocks)</td>
<td>89</td>
</tr>
<tr>
<td>3.47</td>
<td>Real Data vs Laplacian λ = 115 (5 × 5 blocks)</td>
<td>90</td>
</tr>
<tr>
<td>3.48</td>
<td>Real Data vs Laplacian λ = 65 (10 × 10 blocks)</td>
<td>90</td>
</tr>
<tr>
<td>4.1</td>
<td>Prediction Gain (2 × 2 blocks)</td>
<td>92</td>
</tr>
<tr>
<td>4.2</td>
<td>Prediction Gain (5 × 5 blocks)</td>
<td>93</td>
</tr>
<tr>
<td>4.3</td>
<td>Prediction Gain (10 × 10 blocks)</td>
<td>93</td>
</tr>
<tr>
<td>4.4</td>
<td>Prediction Error Comparison (2 × 2 blocks)</td>
<td>94</td>
</tr>
<tr>
<td>4.5</td>
<td>Prediction Error Comparison (5 × 5 blocks)</td>
<td>95</td>
</tr>
<tr>
<td>4.6</td>
<td>Prediction Error Comparison (10 × 10 blocks)</td>
<td>96</td>
</tr>
<tr>
<td>4.7</td>
<td>Prediction Error Autocorrelation (2 × 2 blocks)</td>
<td>97</td>
</tr>
<tr>
<td>4.8</td>
<td>Prediction Error Autocorrelation (5 × 5 blocks)</td>
<td>98</td>
</tr>
<tr>
<td>4.9</td>
<td>Prediction Error Autocorrelation (10 × 10 blocks)</td>
<td>99</td>
</tr>
<tr>
<td>4.10</td>
<td>Filter Cascade</td>
<td>99</td>
</tr>
<tr>
<td>4.11</td>
<td>Long Term Prediction Error Autocorrelation (2 × 2 blocks)</td>
<td>101</td>
</tr>
<tr>
<td>4.12</td>
<td>Long Term Prediction Error Autocorrelation (5 × 5 blocks)</td>
<td>102</td>
</tr>
<tr>
<td>4.13</td>
<td>Long Term Prediction Error Autocorrelation (10 × 10 blocks)</td>
<td>103</td>
</tr>
<tr>
<td>4.14</td>
<td>Prediction Gain vs Frame Size (2 × 2 blocks)</td>
<td>106</td>
</tr>
<tr>
<td>4.15</td>
<td>Prediction Gain vs Frame Size (5 × 5 blocks)</td>
<td>106</td>
</tr>
<tr>
<td>4.16</td>
<td>Prediction Gain vs Frame Size (10 × 10 blocks)</td>
<td>107</td>
</tr>
<tr>
<td>4.17</td>
<td>LP Coefficient Calculation per Frame vs Error Threshold (2 × 2 blocks)</td>
<td>108</td>
</tr>
<tr>
<td>4.18</td>
<td>LP Coefficient Calculation per Frame vs Error Threshold (5 × 5 blocks)</td>
<td>108</td>
</tr>
<tr>
<td>4.19</td>
<td>LP Coefficient Calculation per Frame vs Error Threshold (10 × 10 blocks)</td>
<td>109</td>
</tr>
<tr>
<td>4.20</td>
<td>LP Coefficients Distribution: Actual Data vs Laplacian</td>
<td>111</td>
</tr>
<tr>
<td>4.21</td>
<td>LP Coefficients Quantization (2 × 2 blocks)</td>
<td>112</td>
</tr>
<tr>
<td>4.22</td>
<td>LP Coefficients Quantization (5 × 5 blocks)</td>
<td>112</td>
</tr>
<tr>
<td>4.23</td>
<td>LP Coefficients Quantization (10 × 10 blocks)</td>
<td>113</td>
</tr>
<tr>
<td>4.24</td>
<td>CELP model</td>
<td>114</td>
</tr>
</tbody>
</table>
6.1 Proposed AIRS LP ........................................ 141
6.2 Original vs Reconstructed (PSNR = 33.14 dB), Granule 16 ............ 144
6.3 Original vs Reconstructed (PSNR = 40.77 dB), Granule 16 ............ 144
6.4 Original vs Reconstructed (PSNR = 47.98 dB), Granule 16 ............ 145
6.5 Original vs Reconstructed (PSNR = 60.55 dB), Granule 16 ............ 145
6.6 Original vs Reconstructed (PSNR = 34.54 dB), Granule 82 ............. 146
6.7 Original vs Reconstructed (PSNR = 42.03 dB), Granule 82 ............. 146
6.8 Original vs Reconstructed (PSNR = 49.39 dB), Granule 82 ............. 147
6.9 Original vs Reconstructed (PSNR = 61.66 dB), Granule 82 ............. 147
6.10 Original vs Reconstructed (PSNR = 25.91 dB), Granule 126 ........... 148
6.11 Original vs Reconstructed (PSNR = 34.38 dB), Granule 126 ........... 148
6.12 Original vs Reconstructed (PSNR = 43.08 dB), Granule 126 ........... 149
6.13 Original vs Reconstructed (PSNR = 55.26 dB), Granule 126 ........... 149
6.14 Original vs Reconstructed (PSNR = 27.16 dB), Granule 151 ........... 150
6.15 Original vs Reconstructed (PSNR = 36.05 dB), Granule 151 ........... 150
6.16 Original vs Reconstructed (PSNR = 44.05 dB), Granule 151 ........... 151
6.17 Original vs Reconstructed (PSNR = 56.13 dB), Granule 151 ........... 151
6.18 Original vs Reconstructed (PSNR = 26.25 dB), Granule 193 ........... 152
6.19 Original vs Reconstructed (PSNR = 34.47 dB), Granule 193 ........... 153
6.20 Original vs Reconstructed (PSNR = 43.04 dB), Granule 193 ........... 153
6.21 Original vs Reconstructed (PSNR = 55.31 dB), Granule 193 ........... 154
6.22 Experimental Rate-Distortion (Granule 16) ............................ 154
6.23 Experimental Rate-Distortion (Granule 82) ............................ 155
6.24 Experimental Rate-Distortion (Granule 126) ............................ 155
6.25 Experimental Rate-Distortion (Granule 151) ............................ 156
6.26 Experimental Rate-Distortion (Granule 193) ............................ 156
6.27 Original vs Reconstructed (PSNR = 36.81 dB), Granule 16 (Q = 6) ...... 158
6.28 Original vs Reconstructed (PSNR = 35.59 dB), Granule 82 (Q = 6) ...... 158
6.29 Original vs Reconstructed (PSNR = 40.15 dB), Granule 126 (Q = 6) ...... 159
6.30 Original vs Reconstructed (PSNR = 39.91 dB), Granule 151 (Q = 6) ...... 159
6.31 Original vs Reconstructed (PSNR = 40.11 dB), Granule 193 (Q = 6) ...... 160
6.32 Original vs Reconstructed (PSNR = 44.54 dB), Granule 16 (Q = 7) ...... 160
6.33 Original vs Reconstructed (PSNR = 43.10 dB), Granule 82 (Q = 7) ...... 161
6.34 Original vs Reconstructed (PSNR = 47.22 dB), Granule 126 (Q = 7) ...... 161
6.35 Original vs Reconstructed (PSNR = 47.48 dB), Granule 151 (Q = 7) ...... 162
6.36 Original vs Reconstructed (PSNR = 47.57 dB), Granule 193 (Q = 7) ...... 162
6.37 Original vs Reconstructed (PSNR = 80.61 dB), Granule 16 (Q = 8) . . . . 163
6.38 Original vs Reconstructed (PSNR = 76.87 dB), Granule 82 (Q = 8) . . . . 164
6.39 Original vs Reconstructed (PSNR = 79.24 dB), Granule 126 (Q = 8) . . . . 164
6.40 Original vs Reconstructed (PSNR = 77.85 dB), Granule 151 (Q = 8) . . . . 165
6.41 Original vs Reconstructed (PSNR = 78.08 dB), Granule 193 (Q = 8) . . . . 165
6.42 Original vs Reconstructed (PSNR = 35.70 dB), Granule 16 (Q = 4) . . . . 167
6.43 Original vs Reconstructed (PSNR = 36.67 dB), Granule 82 (Q = 4) . . . . 167
6.44 Original vs Reconstructed (PSNR = 34.56 dB), Granule 126 (Q = 4) . . . . 168
6.45 Original vs Reconstructed (PSNR = 37.76 dB), Granule 151 (Q = 4) . . . . 168
6.46 Original vs Reconstructed (PSNR = 34.86 dB), Granule 193 (Q = 4) . . . . 169
6.47 Original vs Reconstructed (PSNR = 40.89 dB), Granule 16 (Q = 5) . . . . 169
6.48 Original vs Reconstructed (PSNR = 42.51 dB), Granule 82 (Q = 5) . . . . 170
6.49 Original vs Reconstructed (PSNR = 40.26 dB), Granule 126 (Q = 5) . . . . 170
6.50 Original vs Reconstructed (PSNR = 43.59 dB), Granule 151 (Q = 5) . . . . 171
6.51 Original vs Reconstructed (PSNR = 40.69 dB), Granule 193 (Q = 5) . . . . 171
6.52 Original vs Reconstructed (PSNR = 47.59 dB), Granule 16 (Q = 6) . . . . 172
6.53 Original vs Reconstructed (PSNR = 49.25 dB), Granule 82 (Q = 6) . . . . 172
6.54 Original vs Reconstructed (PSNR = 46.97 dB), Granule 126 (Q = 6) . . . . 173
6.55 Original vs Reconstructed (PSNR = 49.99 dB), Granule 151 (Q = 6) . . . . 173
6.56 Original vs Reconstructed (PSNR = 47.27 dB), Granule 193 (Q = 6) . . . . 175
6.57 Original vs Reconstructed (PSNR = 55.60 dB), Granule 16 (Q = 7) . . . . 176
6.58 Original vs Reconstructed (PSNR = 57.97 dB), Granule 82 (Q = 7) . . . . 176
6.59 Original vs Reconstructed (PSNR = 55.57 dB), Granule 126 (Q = 7) . . . . 177
6.60 Original vs Reconstructed (PSNR = 57.76 dB), Granule 151 (Q = 7) . . . . 177
6.61 Original vs Reconstructed (PSNR = 55.43 dB), Granule 193 (Q = 7) . . . . 178
6.62 Original vs Reconstructed (PSNR = 121.84 dB), Granule 16 (Q = 8) . . . . 178
6.63 Original vs Reconstructed (PSNR = 94.05 dB), Granule 82 (Q = 8) . . . . 179
6.64 Original vs Reconstructed (PSNR = 99.61 dB), Granule 126 (Q = 8) . . . . 179
6.65 Original vs Reconstructed (PSNR = 87.04 dB), Granule 151 (Q = 8) . . . . 180
6.66 Original vs Reconstructed (PSNR = 87.28 dB), Granule 193 (Q = 8) . . . . 180
6.67 Original vs Reconstructed (PSNR = 33.25 dB), Granule 16 (Q = 2) . . . . 182
6.68 Original vs Reconstructed (PSNR = 33.24 dB), Granule 82 (Q = 2) . . . . 182
6.69 Original vs Reconstructed (PSNR = 33.20 dB), Granule 126 (Q = 2) . . . . 183
6.70 Original vs Reconstructed (PSNR = 33.19 dB), Granule 151 (Q = 2) . . . . 183
6.71 Original vs Reconstructed (PSNR = 33.14 dB), Granule 193 (Q = 2) . . . . 184
6.72 Original vs Reconstructed (PSNR = 39.38 dB), Granule 16 (Q = 3) . . . . 184

11
6.73 Original vs Reconstructed (PSNR = 39.26 dB), Granule 82 (Q = 3) . . . . 185
6.74 Original vs Reconstructed (PSNR = 39.36 dB), Granule 126 (Q = 3) . . . . 185
6.75 Original vs Reconstructed (PSNR = 39.18 dB), Granule 151 (Q = 3) . . . . 186
6.76 Original vs Reconstructed (PSNR = 39.18 dB), Granule 193 (Q = 3) . . . . 186
6.77 Original vs Reconstructed (PSNR = 44.60 dB), Granule 16 (Q = 4) . . . . 187
6.78 Original vs Reconstructed (PSNR = 44.21 dB), Granule 82 (Q = 4) . . . . 187
6.79 Original vs Reconstructed (PSNR = 44.10 dB), Granule 126 (Q = 4) . . . . 188
6.80 Original vs Reconstructed (PSNR = 44.19 dB), Granule 151 (Q = 4) . . . . 188
6.81 Original vs Reconstructed (PSNR = 44.14 dB), Granule 193 (Q = 4) . . . . 189
6.82 Original vs Reconstructed (PSNR = 47.22 dB), Granule 16 (Q = 5) . . . . 189
6.83 Original vs Reconstructed (PSNR = 46.66 dB), Granule 82 (Q = 5) . . . . 190
6.84 Original vs Reconstructed (PSNR = 46.53 dB), Granule 126 (Q = 5) . . . . 190
6.85 Original vs Reconstructed (PSNR = 46.68 dB), Granule 151 (Q = 5) . . . . 191
6.86 Original vs Reconstructed (PSNR = 46.60 dB), Granule 193 (Q = 5) . . . . 192
6.87 Original vs Reconstructed (PSNR = 47.71 dB), Granule 16 (Q = 6) . . . . 192
6.88 Original vs Reconstructed (PSNR = 47.32 dB), Granule 82 (Q = 6) . . . . 193
6.89 Original vs Reconstructed (PSNR = 47.18 dB), Granule 126 (Q = 6) . . . . 193
6.90 Original vs Reconstructed (PSNR = 47.26 dB), Granule 151 (Q = 6) . . . . 194
6.91 Original vs Reconstructed (PSNR = 47.22 dB), Granule 193 (Q = 6) . . . . 194
6.92 Original vs Reconstructed (PSNR = 40.33 dB), Granule 16 (Q1 = 2, Q21 = 3, Q22 = 4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 196
6.93 Original vs Reconstructed (PSNR = 30.96 dB), Granule 82 (Q1 = 2, Q21 = 3, Q22 = 4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 196
6.94 Original vs Reconstructed (PSNR = 35.47 dB), Granule 126 (Q1 = 2, Q21 = 3, Q22 = 4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 196
6.95 Original vs Reconstructed (PSNR = 35.53 dB), Granule 151 (Q1 = 2, Q21 = 3, Q22 = 4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 197
6.96 Original vs Reconstructed (PSNR = 34.54 dB), Granule 193 (Q1 = 2, Q21 = 3, Q22 = 4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 198
6.97 Original vs Reconstructed (PSNR = 44.90 dB), Granule 16 (Q1 = 3, Q21 = 4, Q22 = 5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 198
6.98 Original vs Reconstructed (PSNR = 36.68 dB), Granule 82 (Q1 = 3, Q21 = 4, Q22 = 5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 199
6.99 Original vs Reconstructed (PSNR = 39.97 dB), Granule 126 (Q1 = 3, Q21 = 4, Q22 = 5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 199
6.100 Original vs Reconstructed (PSNR = 40.50 dB), Granule 151 (Q1 = 3, Q21 = 4, Q22 = 5) .................................................. 200
6.101 Original vs Reconstructed (PSNR = 38.66 dB), Granule 193 (Q1 = 3, Q21 = 4, Q22 = 5) .................................................. 200
6.102 Original vs Reconstructed (PSNR = 51.66 dB), Granule 16 (Q1 = 4, Q21 = 5, Q22 = 6) .................................................. 201
6.103 Original vs Reconstructed (PSNR = 43.55 dB), Granule 82 (Q1 = 4, Q21 = 5, Q22 = 6) .................................................. 201
6.104 Original vs Reconstructed (PSNR = 44.42 dB), Granule 126 (Q1 = 4, Q21 = 5, Q22 = 6) .................................................. 202
6.105 Original vs Reconstructed (PSNR = 43.05 dB), Granule 151 (Q1 = 4, Q21 = 5, Q22 = 6) .................................................. 202
6.106 Original vs Reconstructed (PSNR = 42.79 dB), Granule 193 (Q1 = 4, Q21 = 5, Q22 = 6) .................................................. 203
6.107 Original vs Reconstructed (PSNR = 54.55 dB), Granule 16 (Q1 = 5, Q21 = 6, Q22 = 7) .................................................. 203
6.108 Original vs Reconstructed (PSNR = 45.92 dB), Granule 82 (Q1 = 5, Q21 = 6, Q22 = 7) .................................................. 204
6.109 Original vs Reconstructed (PSNR = 47.60 dB), Granule 126 (Q1 = 5, Q21 = 6, Q22 = 7) .................................................. 204
6.110 Original vs Reconstructed (PSNR = 47.07 dB), Granule 151 (Q1 = 5, Q21 = 6, Q22 = 7) .................................................. 205
6.111 Original vs Reconstructed (PSNR = 46.51 dB), Granule 193 (Q1 = 5, Q21 = 6, Q22 = 7) .................................................. 206
6.112 Original vs Reconstructed (PSNR = 57.21 dB), Granule 16 (Q1 = 6, Q21 = 7, Q22 = 8) .................................................. 206
6.113 Original vs Reconstructed (PSNR = 49.33 dB), Granule 82 (Q1 = 6, Q21 = 7, Q22 = 8) .................................................. 207
6.114 Original vs Reconstructed (PSNR = 49.53 dB), Granule 126 (Q1 = 6, Q21 = 7, Q22 = 8) .................................................. 207
6.115 Original vs Reconstructed (PSNR = 48.95 dB), Granule 151 (Q1 = 6, Q21 = 7, Q22 = 8) .................................................. 208
6.116 Original vs Reconstructed (PSNR = 48.66 dB), Granule 193 (Q1 = 6, Q21 = 7, Q22 = 8) .................................................. 208
6.117 Original vs Reconstructed (PSNR = 42.77 dB), Granule 16 .................................................. 209
6.118 Original vs Reconstructed (PSNR = 42.03 dB), Granule 82 .................................................. 210
6.119 Original vs Reconstructed (PSNR = 43.91 dB), Granule 126 . . . . . . . . . . 210
6.120 Original vs Reconstructed (PSNR = 44.16 dB), Granule 151 . . . . . . . . . . 211
6.121 Original vs Reconstructed (PSNR = 44.25 dB), Granule 193 . . . . . . . . . . 211
6.122 Original vs Reconstructed (PSNR = 46.90 dB), Granule 16 . . . . . . . . . . 212
6.123 Original vs Reconstructed (PSNR = 45.92 dB), Granule 82 . . . . . . . . . . 213
6.124 Original vs Reconstructed (PSNR = 47.60 dB), Granule 126 . . . . . . . . . . 213
6.125 Original vs Reconstructed (PSNR = 47.07 dB), Granule 151 . . . . . . . . . . 214
6.126 Original vs Reconstructed (PSNR = 46.51 dB), Granule 193 . . . . . . . . . . 214
6.127 Compression Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 215
List of Tables

3.1 Dynamic range per band ($d_i$) ........................................... 61
3.2 Mean/Range entropy per granule ......................................... 62
3.3 $\rho(D)$ vs Granule ................................................................. 70
3.4 Bias Adjustment $\xi$ Coefficient ............................................. 76
3.5 Bias Adjustment Rate ............................................................... 76
3.6 Block Scan Analysis ($2 \times 2$ blocks) ...................................... 81
3.7 Block Scan Analysis ($5 \times 5$ blocks) ...................................... 81
3.8 Block Scan Analysis ($10 \times 10$ blocks) ................................. 81
4.1 Long Term Prediction Gain vs Granule ($2 \times 2$ blocks) ............. 104
4.2 Long Term Prediction Gain vs Granule ($5 \times 5$ blocks) ............. 104
4.3 Long Term Prediction Gain vs Granule ($10 \times 10$ blocks) ......... 105
4.4 $k_1$ vs Number of Stages/Block Size ...................................... 105
4.5 Codeword Size ($v$) vs Codebook Size ($U$) ............................. 119
4.6 CELP Compression Rate ($\hat{r}_t$) vs Granule ............................ 122
4.7 Vector Quantization Compression Rate ($\hat{r}_t$) vs Granule .......... 122
5.1 $\lambda_j$ vs Band Ordering/Subband Decomposition ..................... 133
6.1 AIRS LP CELP Parameters ....................................................... 141
6.2 AIRS LP CELP Data (Granule 16) ........................................... 142
6.3 AIRS LP CELP Data (Granule 82) ........................................... 142
6.4 AIRS LP CELP Data (Granule 126) ......................................... 142
6.5 AIRS LP CELP Data (Granule 151) ......................................... 143
6.6 AIRS LP CELP Data (Granule 193) ......................................... 143
6.7 3D APDCM ($Q = 6$) ................................................................. 157
6.8 3D APDCM ($Q = 7$) ................................................................. 157
6.9 3D APDCM ($Q = 8$) ................................................................. 157
6.10 JPEG-7 ($Q = 4$) ................................................................. 163
6.11 JPEG-7 ($Q = 5$) ................................................................. 166
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.12</td>
<td>JPEG-7 (Q = 6)</td>
<td>166</td>
</tr>
<tr>
<td>6.13</td>
<td>JPEG-7 (Q = 7)</td>
<td>166</td>
</tr>
<tr>
<td>6.14</td>
<td>JPEG-7 (Q = 8)</td>
<td>174</td>
</tr>
<tr>
<td>6.15</td>
<td>CALIC (Q = 2)</td>
<td>175</td>
</tr>
<tr>
<td>6.16</td>
<td>CALIC (Q = 3)</td>
<td>181</td>
</tr>
<tr>
<td>6.17</td>
<td>CALIC (Q = 4)</td>
<td>181</td>
</tr>
<tr>
<td>6.18</td>
<td>CALIC (Q = 5)</td>
<td>181</td>
</tr>
<tr>
<td>6.19</td>
<td>CALIC (Q = 6)</td>
<td>181</td>
</tr>
<tr>
<td>6.20</td>
<td>DWT (Granule 16)</td>
<td>191</td>
</tr>
<tr>
<td>6.21</td>
<td>DWT (Granule 82)</td>
<td>195</td>
</tr>
<tr>
<td>6.22</td>
<td>DWT (Granule 126)</td>
<td>195</td>
</tr>
<tr>
<td>6.23</td>
<td>DWT (Granule 151)</td>
<td>195</td>
</tr>
<tr>
<td>6.24</td>
<td>DWT (Granule 193)</td>
<td>195</td>
</tr>
<tr>
<td>6.25</td>
<td>3D SPIHT</td>
<td>205</td>
</tr>
<tr>
<td>6.26</td>
<td>3D SPECK</td>
<td>212</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Multispectral Imaging

*Multispectral* images capture data at different wavelengths across the electromagnetic spectrum. The wavelength range includes light from frequencies well beyond the visible light range. Regular multispectral remote sensors produce images with a few wavelength bands while *hyperspectral* remote sensors (hyper as a synonym of excessive), on the other hand, collect data in tens or hundreds of spectral bands. *Ultraspectral* (ultra meaning beyond hyper) remote sensors generate images with thousands of bands and they are considered the future of spectroscopy [1]. Throughout this dissertation, and unless otherwise indicated, the term multispectral will be used to refer to any generic regular multispectral, hyperspectral or ultraspectral image.

Generic hyperspectral data is typically arranged as a three dimensional structure like the one shown in Figure 1.1 where data consists of two dimensional *band images* that contain the information corresponding to the radiation obtained by a specific detector tuned at a given wavelength.

The remote sensor data can be used to obtain a continuous spectrum for each detection cell in such a way that when applying atmospheric and terrain adjustments to the data, it can be compared to known reflectance spectra in order to interpret the properties of the ground materials that are being measured.

The spectral reflectance is the ratio of reflected energy to incident energy as a function of the wavelength, which varies because the energy at different wavelengths is absorbed or scattered in a different way depending on the ground materials. These variations can be seen in Figure 1.2 where the spectral reflectance curves are shown for different materials.
Figure 1.1: Hyperspectral Data Structure

Figure 1.2: Reflectance Curve
The absorption bands are pronounced downward deflections of the spectral curves since the material selectively absorbs the incident energy. As stated before the overall shape of these curves can be used to classify and identify the materials under observation (i.e. dry soil has a much higher reflectance that wet soil at any of the wavelengths).

Chemical composition as well as crystalline structure control the shape of the spectral reflectance curve in inorganic materials. To be specific particular chemical elements, their ionic charge as well as the geometry of the bonds between them, determine the presence and position of the different absorption bands. Similarly healthy green plants have a characteristic spectral reflectance shape that follows many of the plant features. The shape of the curve is governed by absorption effects from chlorophyll and leaf pigments. In the visible range chlorophyll absorbs green wavelength light less strongly than red and blue wavelength light and as a consequence healthy plants appear green in color. In the near infrared range the reflectance is given by the interaction within the internal structure of the leaves and because these structures change greatly between different species, measuring the spectral reflectance in this range can be used to determine the nature of the vegetation.

The spectral reflectance representation provides an alternative way of viewing hyperspec-
Figure 1.4: Spectral Solar Irradiance

With respect to the solar energy, the solar spectral irradiance is used to determine the...
electromagnetic radiation from the sun incident on the surface of the earth. By looking at
the solar spectral irradiance shown in Figure 1.4 it is clear that the incoming solar energy is
dependent on the wavelength and has its peak on the visible range (between 0.39 \( \mu m \) and
0.75 \( \mu m \)). On the other hand the geometry of the illumination is greatly dependent on the
angle of incidence which is defined as the angle between the path of the incoming energy
and the ground surface. This angle changes with the terrain shape as well as the time of the
day and season. Moreover when topographic features and clouds cast shadows the level of
energy captured by the remote sensor is also affected.

The incoming and reflected energy interacts with the atmosphere and depending on the
wavelength the transmittance is reduced by absorption of certain gases and by scattering
of gas particles as shown in Figure 1.5. Note that at certain wavelengths (i.e. 1.4 \( \mu m \)) the
reflected energy is reduced completely due to water vapor so no useful information can be
obtained for those bands.

It is important to mention that because the remote sensor converts the detected energy
into an electric signal, which is scaled and quantized, an additional set of errors are present at
the moment of digitalizing the data and preparing it for analysis and processing. These are
the sensor effects which require that all raw measurements be scaled to produced comparable results.

1.2 Multispectral Data

In remote sensing applications the availability and capacity of airborne and spaceborne sensors has been increasing considerably throughout the years. Two of the most well known and recognized applications are those carried out by NASA JPL’s hyperspectral AVIRIS and ultraspectral AIRS sensors. In general AVIRIS data is in the visible or near infrared regions with the main purpose of pattern recognition and classification. Under AVIRIS significant data loss from lossy compression is usually acceptable by human visual perception. On the other hand, AIRS data is in the infrared region with the main purpose of retrieving atmospheric temperature, moisture and trace gas profiles, surface temperature and emissivity as well as cloud and aerosol optical properties for better weather and climate prediction [5]. The details of both AVIRIS and AIRS are given in the following subsections.

1.2.1 AVIRIS

AVIRIS stands for Airborne Visible/Infrared Imaging Spectrometer and captures data in the visible and near infrared spectrum in an area of 2 to 12 kilometers wide and several kilometers long into hundreds of non overlapping bands. The captured spectral radiance provides 224 contiguous bands with wavelengths from 0.4 µm to 2.5 µm. Typically the resulting volume of data captured exceeds 500 MB per flight.

The main objective of the AVIRIS project is to identify, measure, and monitor constituents of the Earth’s surface and atmosphere based on molecular absorption and particle scattering signatures. Research with AVIRIS data is predominantly focused on understanding processes related to the global environment and climate change as well as geological mapping, target recognition and anomaly detection.

As an example Figure 1.6 shows a single pixel from AVIRIS where spectral radiance is plotted as a function of the wavelength. The shape is dominated by the light curve of the sun and the absorption features of the atmosphere (which was discussed in the previous section). Those peaks and valleys that are not due to the sun or the atmosphere provide information about the chemical information about the pixel being captured. More information regarding AIRS is available on the NASA AVIRIS website [7].
1.2.2 AIRS

The ultraspectral sounder data can be generated from either a Michelson interferometer (CrIS, IASI and GIFTS) or a grating spectrometer (AIRS). AIRS stands for Atmospheric Infrared Sounder and is the standard reference in compression studies of ultraspectral data. It records thousands of bands covering the infrared spectrum and it is considered one of the most advanced atmospheric sounding systems ever deployed. One of the characteristics of AIRS data is that it shows strong correlation in disjoint spectral regions.

The major application of AIRS is the acquisition of atmospheric parameters such as gases, temperature, moisture of clouds, dust concentrations in addition to other quantities to perform climate and weather forecast.

The AIRS data includes 2378 bands in the 3.74 μm to 15.4 μm range of the mid infrared spectral region. AIRS data obtained throughout a day is divided into 240 granules in such a way that a single granule is captured every 6 minutes. Granules consist of 135 scan lines with 90 pixels per scan line over 2378 bands and therefore there are a total of 28892700 pixels per granule. Given that the quantization level of the data is between 12-bit and 14-bit per pixel depending on the band and because 271 bands are normally ignored (to provide compatibility with other ultraspectral sounders) an average image for a given granule is well
over 40MB in size which amounts to well over 9GB for a single day. Figure 1.7 shows the
digitalized version of a granule representing the reflectance value against the band number
for the first pixel. More information regarding AIRS is available on the NASA AIRS website
[6]. AIRS data can be found available for downloading at the AIRS website [8].

1.2.3 AIRS Processing

As previously indicated the primary data returned by AIRS is the infrared spectrum in
individual frequencies. The infrared spectrum is rich in information on numerous gases in
the atmosphere. It forms a “fingerprint” of the state of the atmosphere for a given time and
place that can be used as a climate data record for future generations.

The AIRS Science Processing System (SPS) is a collection of programs, or Product
Generation Executives (PGE), used to process AIRS Science Data. These PGE process
raw, low level AIRS Infrared instrument data to obtain temperature and humidity profiles.
AIRS PGE can be grouped into four distinct processing phases for processing: Level 1A,
Level 1B, Level 2 and Level 3. Each consecutive processing phase yields a higher-level
data product. Levels 1A and 1B result in calibrated, geolocated radiance products. Level
2 processing derives temperature and humidity profiles, and cloud and surface properties. Level 3 produces gridded ascending and descending products from the Level 2 products averaged daily, over 8-day periods and for each calendar month [9].

AIRS data processing begins with receipt of Level 0 data from the Earth Observing System (EOS) Data and Operations System (EDOS). When Level 0 data are received, Level 1A PGE are scheduled. The Level 1A PGEs perform basic house keeping tasks such as ensuring that all the Level 0 data are present and ordering the data into time of observation synchronization. Once the Level 0 data are organized, algorithms perform geolocation refinement and conversion of raw Data Numbers to Engineering Units (DN to EU). Finally, the level 1A data are collected into granules of data and are forwarded to Level 1B PGE for further processing.

Level 1B PGEs receive 240 granules of AIRS Level 1A EU data and produce calibrated, geolocated radiance products. Calibration data and calibration control parameters are analyzed to develop processing specifications for Level 1B. Similarly the single Level 2 PGE reads corresponding Level 1B data granules from all instruments, the surface pressure and a digital elevation map. Depending upon the results of tests applied at each stage of the retrieval flow, the resulting Level 2 product may be one of several possibilities. Level 3 products are statistical summaries of geophysical parameters that have been spatially and temporally re-sampled from lower level data products. Due to re-sampling and selecting a reduced set of reporting parameters, Level 3 datasets are substantially smaller than the lower level source products from which they are derived.

1.3 Ultraspectral Data Compression

As mentioned in Section 1.2.2 ultraspectral images imply huge volumes of data that in many cases exceed available transmissions bandwidths. In fact in order to improve the real-time distribution of data and minimize the storage at base stations, efficient ultraspectral data compression becomes a key issue. In addition to the high correlation in the spatial dimension (common to most natural images) ultraspectral images exhibit similar spectral signatures in adjacent locations.

1.3.1 Lossless vs Lossy Compression

Compression techniques fall into two types: lossless and lossy compression based on whether the image is fully regenerated using the compressed data [3]. Under lossless compression the
regenerated image is identical to the original one and compression is accomplished from a more efficient storage of the information. Under lossy compression some non essential (not relevant to the analysis of the image) information is discarded improving the compression rate but affecting the overall quality of the image.

Lossy compression is extremely dependent on the distortion measure which is used as the decision maker to discard the non essential symbols while ensuring quality. Note that depending on the specific application of the ultraspectral data different distortion measures may be needed. For example a distortion measure that may be suitable for target detection may not work with material identification. However in most cases the application is unknown at the time of compression and therefore a more generic distortion measure is required. In general for a given distortion measure and an a specific level of distortion it is desirable to predict the performance of the algorithm based on the original data before performing any compression.

### 1.3.2 Distortion Measure

The following list summarizes some of the most important distortion measures used by the compression techniques [2].

- **Mean Square Error (MSE)**: Most common distortion measure, it is simple but does not correlate well with subjective ratings.

- **Maximum Absolute Distortion (MAD)**: It guarantees that every reconstructed pixel \( \hat{\omega}(x, y, t) \) is within a maximum distance \( d \) of its original value \( \omega(x, y, t) \).

- **Percentage Maximum Absolute Distortion (PMAD)**: It is like MAD but the distance is measured as a percentage of the original value.

In the case of AIRS images, which are Level 1B radiance maps (as defined in Section 1.2.3), the accuracy of higher Level AIRS products can be used to evaluate the distortion introduced by the compression algorithm itself. Specifically in this dissertation, and based on the International MODIS/AIRS Processing Package (IMAPP) developed by the Cooperative Institute for Meteorological Satellite Studies (CIMSS) at University of Wisconsin [10], the following AIRS Level 2 products are evaluated (with units shown between brackets whenever available).

- Surface Pressure [hPa]
- Surface Skin Temperature [K]
- Atmospheric Temperature [K]
- Total Precipitable Water [cm]
- Atmospheric Moisture [g/kg]
- Total Ozone Amount [dobson units]
- Atmospheric Ozone [ppmv]
- Surface Emissivity
- Surface Reflectivity

The Mean Percentage Error (MPE) between the Level 2 products derived from both the reconstructed and original AIRS images is used to evaluate the compression algorithm performance. This metric is defined as

$$\text{MPE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|r_t - o_t|}{o_t}$$

where $n$ is the number of samples and $r_t$ and $o_t$ are the reconstructed and original samples respectively.

Throughout this document Peak Signal to Noise Ratio (PSNR) is used as the primary distortion measure, specifically the PSNR is calculated as

$$\text{PSNR} = 10 \log_{10} \left( \frac{\text{MAX}_1^2}{\text{MSE}} \right)$$

where $\text{MAX}_1$ is the maximum possible pixel value and it MSE is the mean square error.

Structural Similarity (SSIM) is another alternative metric that can be used to measure the similarity between images by means of comparing local patterns of pixel intensities and taking into account the strong interdependency of pixels that are spatially close [57]. SSIM is calculated in various windows of $8 \times 8$ distributed throughout the image and the resultant index is a value between $-1$ and $1$ where $1$ is only reachable in case both data sets are identical. Since multiple SSIM coefficients are obtained for a single band/image the Mean Structural Similarity (MSSIM) is used instead. The measure between two windows $x$ and $y$
of common size $N \times N$ is given by

$$SSIM = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

where $\mu_x$, $\mu_y$, $\sigma_x$ and $\sigma_y$ are the mean and variances of $x$ and $y$ respectively, $\sigma_{xy}$ is the covariance of $x$ and $y$ and $c_1$ and $c_2$ are two variables to stabilize the division with weak denominator.
1.3.3 Architecture for Ultraspectral Data Compression

The basic architecture for ultraspectral data compression has two components: Pre/Post processing and Compression/Reconstruction [2]. The typical encoder and decoder are shown in Figures 1.8 and 1.9 respectively. Both components are described below:

- Pre/Post processing: The encoder applies some basic reversible process that generates processing side information (unknown to the decoder) beforehand. The decoder reverses the process and applies the same side information to recover the data processed by the encoder. Note that the processing side information is sent to the decoder along with the compression side information and the compressed data itself (which is described below). It will be seen in Chapter 2 that typical processing stages for ultraspectral images are band reordering and sample normalization.

- Compression/Reconstruction: The encoder applies some image compression technique to the original data and based on the desired rate-distortion (a result of the distortion measure) it defines the compression parameters as compression side information which are sent to the decoder along with the compressed data resulting from the compression. The decoder uses the compression side information as well as the compressed data to reconstruct the original image. Unless it is a lossless compression, distortion is introduced in the decompressed data and the process is not fully reversible. As it will be seen in Chapter 2 typical compression stages for ultraspectral images are linear prediction, vector quantization and transform coding.

From this point on in this dissertation, and unless it is specifically indicated, whenever preprocessing and compression is mentioned the content of the discussion also applies to post processing and reconstruction respectively.

1.4 Objectives and Contribution

The goal of this dissertation is to present a whole new architecture for lossy compression of ultraspectral data (more specifically AIRS data) using sound mathematical models that can be used to describe a set of algorithms and their practical implementation. The objectives to be addressed by this dissertation are the following:

- Present a new preprocessing reversible stage that will rearrange the ultraspectral data to make it more efficient when the compression stage is performed.
• Present a new compression stage that will improve the compression rate for a given distortion when compared to current ultraspectral data compression techniques. This involves defining a distortion measure and its effect on real ultraspectral applications.

• Define a mathematical model to approximate the rate-distortion of the compression algorithm and compare it against the real performance of the proposed algorithm for AIRS data.

• Evaluate the performance of the overall architecture and algorithms as well as discuss possible optimizations.

This dissertation contributes to the current state of the art in regards to ultraspectral image compression by describing a novel approach based on full integration of preprocessing and compression stages. This new architecture accomplishes rate-distortion ratios that are far superior to those of other existent compression techniques applied to the same images. In addition the experimental results are validated by means of a new mathematical model that is used to successfully estimate the rate-distortion values.

The following chapters of this dissertation are organized as follows: Chapter 2 provides an exhaustive review of the current ultraspectral image compression techniques. A detailed description of the new preprocessing and compression stages is presented in Chapters 3 and 4 respectively. The corresponding performance of these stages is mathematically justified by means of a rate-distortion analysis in Chapter 5. Experimental results as well as comparison to other compression techniques are given in Chapter 6. Finally Chapter 7 summarizes the conclusions of this dissertation and introduces possible future work related to this novel architecture.
Chapter 2

Review of Literature

2.1 Basic Techniques

For both preprocessing and compression stages, there are many different techniques that are the basis of hyperspectral and ultraspectral image compression. A quick summary of these basic techniques will be presented before providing a review of the current literature. Note that many of them are widely used in generic lossy data and media compression.

2.1.1 Entropy Coding

*Entropy coding* involves converting symbols (for example mapping an ultraspectral image) into a stream of binary codewords. It is normally part of the compression stage and it can be categorized into different types. The most common ones are described below:

- **Huffman Coding**: Higher probability symbols get shorter codewords. Codewords are fixed and cannot easily adapt [11].

- **Arithmetic Coding**: It produces a single codeword that describes a particular combination of messages. Recalculates the statistics of the data during the coding process (known to both encoder and decoder so no extra side information is required) [11].

- **Golomb Coding**: The Golomb coding of parameter $m$ codes $n$ by encoding $n \mod m$ binary and $n \div m$ unary (unary of $n$ is a string of $n$ 0’s and a 1). When $m = 2^k$ the Golomb codes are called Rice codes [13].
Note that the are many variations of these basic types, for example it is possible to have a dynamic version of Huffman coding where codebooks are recalculated on both encoder and decoder as symbols are processed. For most cases entropy coding is fully reversible.

2.1.2 Vector Quantization

*Vector quantization* (VQ) is typically applied to the compression stage and like most quantization techniques is not fully reversible as it introduces distortion [22]. Vector quantization is particularly appealing to the coding of multispectral imagery because pixel vectors have components that are naturally highly correlated. Furthermore pixel based compression simplifies some operations like browsing, classification and target detection. In general the vector quantization process involves the following stages:

- **Vector Formation**: The data (i.e. either preprocessed or multispectral data) is divided into vectors.
- **Set Generation**: A subset of the most representative data is used as the *training set*.
- **Codebook Generation**: A *codebook* is generated from the training set. An algorithm such as LBG [23] is used for this task and a set of codewords or codebook vectors is obtained.
- **Quantization**: Original data is tested against the codebook and the index of the closest codebook vector is used to represent the original sample.

Vector quantization treats a whole set of scalars as a vector and the distortion is minimized by finding the best vector, that is, the best combination of individual scalars. Therefore the use of VQ is far more efficient than regular scalar quantization. Since VQ is inherently expensive to implement, most studies on VQ have been focused on cost reduction. In order to lower the implementation costs, a certain structure can be imposed to reduce the space requirement and/or the time requirement. *Multistage VQ* (MSVQ) provides substantial savings in both space and time but it suffers performance degradation when compared to an equivalent VQ scheme. A simpler method is *Split VQ* where coding a high-dimensional vector is done by splitting the vector into two or more subvectors [24].
2.1.3 Transform Coding

Under *Transform Coding* original sample values are multiplied by a set of *basis vectors* and the product values are added together to generate the transformation coefficients. Since the process is reversible the same procedure is executed on both the encoder and the decoder. Note that transform coding is used in either the preprocessing or the compression stages. Since the transformation coefficients indicate the frequency content of the original waveform highly correlated samples cause high frequency coefficients to be zero. In general for the process to be reversible the basis vectors \( t \) are required to be orthonormal:

\[
    t'_p t_q = \begin{cases} 
    0 & p \neq q \\
    1 & p = q 
\end{cases}
\]

For multispectral image processing three different types of transform coding are usually used: *PCA*, *DCT* and *DWT*. These transformations are described in the following sections.

2.1.3.1 PCA

The *Principal Component Analysis* (PCA) is the most efficient way of maximizing the amount of energy contained in the smallest number of coefficients. Under PCA the basis vectors are the eigenvectors of the covariance matrix (columns of \( Q \)). \( \Delta \) is the diagonal matrix of the eigenvalues of \( A \) [25].

\[
    QA = Q\Delta
\]

PCA performs an axis rotation where the coefficients which correspond to the direction of minimal variance can be discarded. Figure 2.1 describes this process where the data in (b) has a smaller horizontal variance that the same data in (a). Since the basis vectors need to be transmitted as side information this transformation is data dependent.

2.1.3.2 DCT

The *Discrete Cosine Transform* (DCT) basis vectors are almost identical to the basis vectors that would be produced by PCA but because DCT uses a fixed set of basis vectors it is a suboptimal transformation (there is no need to transmit basis vectors as side information) [26]. The DCT basis vectors are given by
Many of the state of the art image and video compression standard are DCT based. It has been extended to the tridimensional plane in order to support multispectral imaging.
2.1.3.3 DWT

Discrete Wavelet Transform (DWT) is a subband filtering process as the one shown by Figure 2.2 [11].

The basis vectors are subsampled and shifted versions of the impulse response of the filter used in the subband filtering process. A two channel decomposition can be repeated on the low pass subbands of the previous stage to provide a multiresolution decomposition. Note that the unidimensional approach can be extended to multiple dimensions to address the image compression problem.

2.1.4 Linear Mixture Analysis

Linear Mixture Analysis (LMA) assumes that the data contains a number of pure pixel vectors (referred as end members) and all other pixel vectors can be expressed as a linear combination of the end members plus a noise [27]. The mixture follows expression (2.1).

\[ x = M\alpha + n \]  

(2.1)

where \( M = [t_1 t_2...t_p] \) represents the pure pixel vectors, \( \alpha = [\alpha_1 \alpha_2...\alpha_p]' \) is the abundance fraction for the corresponding end members and \( n \) is the column vector of noise values.

Linear mixture analysis achieves compression by approximating a single pixel vector by a linear combination of a small number of end members.

2.1.5 Linear Prediction

Under Linear Prediction spatial and spectral data sent to the decoder is used to predict the current value as shown in Figure 2.3. The prediction error defined as the difference between the predicted and the original signal (known as prediction residuals or prediction error) is
sent to the decoder for the decoder to reconstruct the original samples. In general linear prediction is used in the compression stage, and depending on the techniques, the filter coefficients used for the prediction are sent as side information along with the prediction error.

For the most general one dimensional case the basic assumption is that the data can be modeled as an autoregressive (AR) signal and that linear prediction allows redundancy removal in such way that information repeated in an event is eliminated [29]. Specifically linear prediction is described as a system identification problem, where the parameters of an AR model are estimated from the signal itself as it is shown in figure 2.4. The white noise signal $x_n$ is filtered by the AR process synthesizer to obtain $s_n$ with the AR parameters denoted by $\hat{a}_n$. Under the AR model the current sample is predicted as a combination of $L$ past samples as shown by equation (2.2). Note that $L$ is the prediction order of the filter.

\[
\hat{s}_n = -\sum_{i=1}^{L} a_i s_{n-i}
\]  

(2.2)

The $a_i's$ are the estimates of the AR parameters and are referred as the linear prediction (LP) coefficients. The prediction error, shown by equation (2.3), is the difference between the prediction and the original signal and tends to be approximately a white noise signal with a power level that decreases as the prediction order increases.

\[
e_n = s_n - \hat{s}_n
\]  

(2.3)

The optimal LP coefficients can be calculated by minimizing the mean-squared prediction
error as shown by equations (2.4) and (2.5).

$$J = E \left[ e_n^2 \right]$$  \hspace{1cm} (2.4)

$$\frac{\partial J}{\partial a_k} = 2E \left( s_n + \sum_{i=1}^{L} a_i s_{n-i} \right) s_{n-k} = 0$$  \hspace{1cm} (2.5)

This leads to LP coefficients that follow the normal equation

$$R \cdot a = -r$$  \hspace{1cm} (2.6)

with R, a and r as shown below.

$$R = \begin{bmatrix}
R_0 & R_1 & \cdots & R_{L-1} \\
R_1 & R_0 & R_1 & \cdots \\
\vdots & \ddots & \ddots & \ddots \\
R_{L-1} & R_{L-2} & R_{L-3} & R_0
\end{bmatrix}
\quad a = \begin{bmatrix}
a_1 \\
\vdots \\
a_L
\end{bmatrix}
\quad r = \begin{bmatrix}
R_1 \\
\vdots \\
R_L
\end{bmatrix}$$

For all these expressions $R_l$ is the autocorrelation of the $s_n$ signal.

Since the signals to be compressed are nonstationary, the autocorrelation values must be estimated and changed for every short interval of time, that is, their values are recalculated in each frame. Two procedures exist: nonrecursive and recursive. In the nonrecursive approach the window used for extraction has a finite length, while an infinite-length but exponentially decreasing window (which effectively is finite-length) is used for recursive methods. The use of either of these techniques depends on the particular application.

Nonrecursive methods are based on a well defined window sequence $w_n$ that is used to extract the signal of interest for further processing. The Hamming window being one of the most widely used and it is defined as [32]

$$w_n = \begin{cases}
0.54 - 0.46 \cos \left( \frac{2\pi n}{V-1} \right) & 0 \leq n \leq V - 1 \\
0 & n < 0, n > V - 1
\end{cases}$$

with $V$ being the window length.

For a signal frame that ends at time instant $m$ and using the frame length $V$ the nonrecursive method yields to the following autocorrelation.
\[ R_{l,m} = \frac{1}{V} \sum_{n=m-V+1+|l|}^{m} x_n w_{m-n} x_{n-|l|} w_{m-n+|l|} \]

It can be shown that \( R_{l,m} \) is a biased estimator of \( R_l \).

For certain applications it might be necessary to perform the calculation of the autocorrelation in an interval that is shorter and where stationarity is not strong. In this case the autocorrelation can be estimated by applying the nonrecursive method to a larger overlapped version of the original signal (repeating samples). In order to optimize this procedure the recursive approach can be used instead. In the recursive case, information from the past frames is used to update the estimates of the present frame so as to increase the efficiency [30]. The recursive estimator of the autocorrelation can be calculated as

\[ R_{l,m} = \sum_{n=-\infty}^{\infty} x_n w_{m-n} x_{n-t} w_{m-n+t} \]

where \( w_n \) is the Barnwell window. The Barnwell window is defined as

\[ w_n = (n + 1)\alpha^n u_n \]

Figure 2.5 shows the Barnwell window for different values of the parameter \( \alpha \) which controls the window length since the magnitude of the window is negligible outside a certain finite length interval.

There are two ways to determine the filter coefficients, one way is by means of forward linear prediction where the LP coefficients are calculated for the signal which is being predicted. The other way is by means of the backward linear prediction where the LP coefficients are calculated based on samples already decoded.

Figures 2.6 and 2.7 show the block diagrams of the AR process encoder and decoder respectively. As it can be seen the predictor with the difference equation (2.2) is utilized. These block diagrams generate the exact same equations for the AR model.

There is another model moving averages (MA) that is the dual of the AR model. Figures 2.8 and 2.9 show the predictor-based block diagrams of the encoder and decoder filters for MA. In this case the difference equation of the predictor is given by equation (2.7) [29].

For both models Figures 2.10 through 2.13 show the block diagrams of the encoder and decoder when error quantization is introduced.
Figure 2.5: Barnwell window

Figure 2.6: AR encoder
Figure 2.10: AR encoder

Figure 2.11: AR decoder
Figure 2.12: MA encoder

Figure 2.13: MA decoder
\[ \hat{s}_n = -\sum_{i=1}^{L} b_i e_{n-i} \quad (2.7) \]

The \( b_i \)'s are the MA parameters and \( e_n \) is the prediction error signal. When compared to equation (2.2) it can be seen that the prediction is now based on a linear combination of samples of the prediction error which in theory is white noise. Unlike the AR model in which the optimal parameters can be found by solving a set of linear equations, based on the statistics of the observed signal, the MA coefficients can only be found using a set of non-linear equations which are in practice computationally very demanding to solve. As part of this dissertation the implications of MA based linear prediction will be investigated.

### 2.2 Preprocessing Techniques

This section summarizes the most common preprocessing techniques used by multispectral imaging.

#### 2.2.1 Band Ordering

*Band Ordering* is a procedure by which the bands of a multispectral image are arranged to increase the similarity between them and improve compression when linear prediction is used in the compression stage [31]. The similarity between bands \( X \) (with mean \( \hat{X} \)) and \( Y \) (with mean \( \hat{Y} \)) is measured using the correlation given by equation (2.8).

\[
\rho_{X,Y,D} = \frac{\sum_{j=0}^{M} \sum_{i=0}^{N} (X_{j,D,i,D} - \hat{X}) (Y_{j,D,i,D} - \hat{Y})}{\sqrt{\sum_{j=0}^{M} \sum_{i=0}^{N} (X_{j,D,i,D} - \hat{X})^2 \sum_{j=0}^{M} \sum_{i=0}^{N} (Y_{j,D,i,D} - \hat{Y})^2}} \quad (2.8)
\]

Note that \( M \) and \( N \) are the width and height of a given band and \( D \) defines the computation speed in such way that larger \( D \) corresponds to faster calculation. It is easy to see that the ordering problem is similar to the problem of finding a *minimum spanning tree* (MST) in weighted graphs. Vertices are image band members and edges indicate whether there is a direct connection or not. The output is a set of bands that corresponds to how the bands should be arranged.
2.2.2 Bias Adjusted Reordering

Under Bias Adjusted Reordering (BAR) tridimensional data (such as multispectral images) are made bidimensional by converting bidimensional bands into unidimensional signals via continuous scan (progressive, linear, zig-zag or peano) that smooths the transitions [54]. For example a tridimensional image of dimension $n_c n_x n_y$ ($n_c$ is the number of bands, $n_x$ is the band width and $n_y$ is the band height) is reshaped bidimensional as $n_c n_s$ where $n_s = n_x n_y$. BAR can be applied to spatial and/or spectral components to explore the correlation of disjoint geographical regions [16].

The algorithm works as follows: Given a pool of vectors ($S$) not yet reordered and a reference vector, each vector in $S$ is bias adjusted by a constant $b$ for better match with the reference. The best match is the nearest neighbor of the reference vector. Its associated vector is removed from $S$ and it becomes the new reference vector. The process is repeated until $S$ is empty.

Bias coefficients and new band ordering data are encoded as side information. Then regular bidimensional image compression is applied to the bias adjusted multispectral image.

2.3 Compression Techniques

This section summarizes the most common compression techniques used by multispectral imaging.

2.3.1 Entropy Coding

The basic types of entropy coding described in Section 2.1.1 are normally used together with the multispectral image compression techniques discussed throughout this chapter. There also exist some other entropy coding mechanisms such as Tunstall coding which is a variable-to-fixed length encoding algorithm. In the codebook tree each branch denotes a symbol seen and each node is associated with a probability of having seen the symbols along the path from the root to the node. An unexpanded set of nodes is recorded whose initial only element is the root of the tree. To grow the tree one step the node having the highest probability is selected and split into a number of nodes each branch corresponding to a symbol following the parent node. The probability of a new node is the product of its parent node probability by the branch symbol probability. The unexpanded set will contain all the leaf nodes of the tree. Each leaf node corresponds to a Tunstall codeword. The growing steps will continue until
a desired number of leaf nodes are reached in the unexpanded set for codeword assignment. Tunstall coding has been used with linear prediction of AIRS images [21].

2.3.2 Vector Quantization

To minimize the effects of a wide dynamic range in multispectral images the normalization (as part of the preprocessing stage) of the input vector is normally required to improve the performance of the quantizer. Many techniques deal with the way in which the normalization is performed as it is described in the following summary.

- **Mean Residual Vector Quantization (M/RVQ):** Vector mean is subtracted from the input vector [33].

- **Normalized Vector Quantization (NVQ):** Input vectors are converted to zero mean and unit standard deviation [34].

- **Mean Normalized Vector Quantization (MNVQ):** Input vectors are normalized based to the mean of each vector [34].

Other techniques have to do with the classification speed of vector quantization such like **Multiple Subcodebook Algorithm (MSCA)** where the image is spatially segmented and codebooks local to each spatial region are used.

One problem with vector quantization is that because the dimension of the pixel vectors in the current imagers is typically large the algorithms designed to generate codebooks become prohibitively inefficient. One solution to this problem is to divide the pixel vectors into non overlapping subvectors and encode each of them with an independent vector quantizer of reduced dimensionality by means of **Partitioned Vector Quantization (PVQ)** [17]. This approach is further improved with **Locally Optimal Vector Quantization (LOVQ)** in such a way that the segmentation of the vectors is MSE optimal [18].

**Lattice Coders** have been used to quantize prediction error of AIRS data [39]. When the input is a stream of samples from a Gaussian random variable, a fixed number $k$ of samples can be collected into a $k$ dimensional vector for vector quantization. As $k$ grows most of the vectors formed will tend to fall on a $k$ dimensional sphere with radius $\sigma\sqrt{k}$ where $\sigma$ is the standard deviation of the Gaussian random variable. To keep the quantization error small a huge codebook is often required to distribute the code points uniformly on the sphere. But as the size of the codebook grows transmission of the codebook becomes a burden. Lattice
codes can be used to avoid sending a huge codebook since code points in the codebook can be generated from the regular structures of the lattice by a few parameters. Using a subset of those lattice points lying on a sphere centered at the origin as code points, a high dimensional vector of Gaussian random numbers can be represented with less quantization error.

2.3.3 Transform Coding

For each of the transform coding techniques described in the previous section there exist specific applications to multispectral imaging. For most of them some sort of entropy coding is used to encode the transformation coefficients.

2.3.3.1 PCA

Many multispectral compression techniques involve coupling a decorrelating transform in the spectral direction with a DWT in the spatial direction. It has been shown that PCA performs better than any other common use transform in the spectral dimension [14].

2.3.3.2 DCT

Under DCT lossless and near lossless compression can be obtained by extending the DCT to a tridimensional space. As in the standard bidimensional imaging cases arithmetic and Huffman entropy coding is used [35].

2.3.3.3 DWT

Wavelet coefficients are represented by bit planes with most-significant-bits coded first. Depending on the entropy coding type used to encode the wavelet transform coefficients, there are two main groups zero tree coding and context-based coding.

- Zero Tree Coding (DWT-ZTC): Under zero tree coding bit values of each bit plane are coded in order from the lowest to the highest frequency band. Bits in the same spatial position of each band are combined to form a tree of bits providing good compression since high frequency branches are normally zero. Set Partitioning in Hierarchical Trees (SPIHT) and Set Partitioned Embedded Block (SPECK) and their tridimensional versions are the most well known implementations [36].
• **Context-Based Coding (DWT-CBC):** Under context-based coding a specific bit is coded based on the adjacent bit values in the current bit plane and in more significant bit planes. Regions of the image containing low frequency information will produce zero bit in all bit planes. The standard *Joint Photographic Expert Group 2000* (JPEG2000) provides the most well known implementation of this coding [11].

### 2.3.4 Fractal Based Coding

The *fractal based* (FB) compression scheme is inspired by the theory of fractal geometry and has been developed for still image coding. It is performed in the waveform domain and it is implemented with less complexity than any other method. Specifically the bidimensional (regular) or tridimensional (multispectral) image is converted to the wave domain by means of a peano scanning and FB compression is applied to the resulting signal.

The basic idea of the FB compression scheme is that a fixed length *yardstick* is chosen to traverse the image intensity waveform as show in figure 2.14 in such a way that the yardstick length is $y$ and the image intensity at value $x_i$ is $g_i$. One end of the yardstick is placed at the point $(x_i, g_i)$ and the other end intersects the waveform at $x_{i+1}$. Though the intersection may not be unique, the algorithm makes a selection by picking the nearest such point. In
this method, the sign bit, corresponding to the descent or ascent of the waveform and the projected distance \( t_i \) are transmitted and used to reconstruct the original signal [54].

### 2.3.5 Linear Prediction

Since ultraspectral images exhibit great correlation not only in the spatial dimension but also in the spectral one they become great candidates for linear prediction. This section summarizes some of the most important techniques related to linear prediction.

#### 2.3.5.1 3D ADPCM

Under 3D Adaptive Differential Pulse Code Modulation (3D ADPCM) interband and intra-band correlation is exploited [51]. For each band a different predictor taken from those shown in Figure 2.15 is selected. Note that five casual predictors for predictive decorrelation are used. \( SA \) indicates a pure spatial predictor, \( SE \) indicates a pure spectral predictor and \( SS \) indicates a mixed spatial-spectral predictor.

![Table: Predictors and Formulas](image)

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Formula ( \hat{x}_{i,j,\lambda} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA-2RC</td>
<td>( \frac{x_{i-1} + x_{i,j-1,\lambda}}{2} )</td>
</tr>
<tr>
<td>SS-1</td>
<td>( x_{i,j-1,\lambda} + x_{i,j,\lambda-1} - x_{i,j-1,\lambda-1} )</td>
</tr>
<tr>
<td>SE-o1B</td>
<td>( a + bx_{i,j,\lambda-1} )</td>
</tr>
<tr>
<td>SE-o2B</td>
<td>( a + bx_{i,j,\lambda-1} + cx_{i,j,\lambda-2} )</td>
</tr>
<tr>
<td>SS-o1</td>
<td>( a + bx_{i,j-1,\lambda} + cx_{i,j,\lambda} + dx_{i,j-1,\lambda-1} )</td>
</tr>
</tbody>
</table>

**Figure 2.15: 3D ADPCM**

The predictor selection is based on the least squares estimate \( \sum_{j=1}^{n} (\hat{x}_{i,j,\lambda} - x_{i,j,\lambda})^2 \) where \( x_{i,j,\lambda} \) is the pixel at position \( i, j \) on band \( \lambda \). The prediction residuals are encoded using Rice coding.

#### 2.3.5.2 JPEG-7

Joint Photographic Expert Group - 7 (JPEG-7) is part of the JPEG standard for encoding of bidimensional images using linear prediction [3]. It is based on seven schemes as shown...


\[ \hat{I}_{i,j} = I_{i-1,j} \]

\[ \hat{I}_{i,j} = I_{i,j-1} \]

\[ \hat{I}_{i,j} = I_{i-1,j-1} \]

\[ \hat{I}_{i,j} = I_{i,j-1} + I_{i-1,j} - I_{i-1,j-1} \]

\[ \hat{I}_{i,j} = \frac{I_{i,j-1} + I_{i,j-1} - I_{i-1,j-1}}{2} \]

\[ \hat{I}_{i,j} = \frac{I_{i,j-1} + I_{i,j-1} - I_{i-1,j-1}}{2} \]

\[ \hat{I}_{i,j} = \frac{I_{i-1,j} + I_{i,j-1} - I_{i-1,j-1}}{2} \]

\[ \hat{I}_{i,j} = I_{i,j-1} + I_{i,j} \]

Figure 2.16: JPEG-7

```
1  \hat{I}_{i,j} = I_{i-1,j}
2  \hat{I}_{i,j} = I_{i,j-1}
3  \hat{I}_{i,j} = I_{i-1,j-1}
4  \hat{I}_{i,j} = I_{i,j-1} + I_{i-1,j} - I_{i-1,j-1}
5  \hat{I}_{i,j} = \frac{I_{i,j-1} + I_{i,j-1} - I_{i-1,j-1}}{2}
6  \hat{I}_{i,j} = \frac{I_{i,j-1} + I_{i,j-1} - I_{i-1,j-1}}{2}
7  \hat{I}_{i,j} = \frac{I_{i-1,j} + I_{i,j-1} - I_{i-1,j-1}}{2}
```

Figure 2.17: JPEG-LS
by Figure 2.16. The image is segmented into regions and all predictors are tried but the one with the best compression is used. The whole procedure can be extended to multispectral images by running all the schemes against a reference band.

### 2.3.5.3 CALIC

Context-based Adaptive Lossless Image Codec (CALIC) is a multispectral compression technique where based on local gradients a specific prediction scheme is selected. Any bias in the prediction is removed and the prediction error is encoded using Huffman or arithmetic coding based on 8 contexts derived from the gradients. When applying CALIC to multispectral images, the coefficient $\phi$, shown by equation (2.9) can be obtained to determine whether intraband or interband prediction must be performed. Specifically for a given pixel $(\hat{x}, \hat{y})$, $\phi$ is calculated using pixels $(x_i, y_i)$ ($0 \leq i \leq n$) in the current or previous band depending on whether intraband or interband prediction is being evaluated.

$$\phi = \frac{\sum_{i=0}^{n} [(x_i - \hat{x})(y_i - \hat{y})]}{\sqrt{\sum_{i=0}^{n} [(x_i - \hat{x})^2(y_i - \hat{y})^2]}} \quad (2.9)$$

CALIC and BAR (as described in Section 2.2.2) have been combined in such a way that the tridimensional AIRS data is converted to a bidimensional plane via BAR and compressed using CALIC [16]. BAR is performed on spectral, spatial and spectral-spatial dimensions. In general the CALIC-BAR combination performs better than the CALIC algorithm alone for 10 generic granules of AIRS data.

### 2.3.5.4 JPEG-LS

Joint Photographic Expert Group - LS (JPEG-LS) uses non-linear prediction [3] with the predicted value given by

$$\hat{p}_{i,j} = \begin{cases} 
\text{min}(N,W) & \text{if } NW < \text{max}(N,W) \\
\text{max}(N,W) & \text{if } NW < \text{min}(N,W) \\
N + W - NW & \text{otherwise}
\end{cases}$$

This estimator is known as the Median Edge Detector (MED) and the coordinates are as shown in figure 2.17. The context formation is given by $D_1 = NE - N$, $D_2 = N - NW$ and $D_3 = NW - W$ where $D_1$, $D_2$ and $D_3$ are quantized as $q_1$, $q_2$ and $q_3$ and used to estimate $p(e|q_1, q_2, q_3) = p(-e|q_1, q_2, q_3)$. The residuals are encoded using Rice coding.
2.3.5.5 CELP

Grouping representative error signals for each speech frame in a dictionary leads to the *Code Excited Linear Prediction* (CELP) approach. In this case a codebook with a large number of codewords is used for reference to generate the error signal allowing the transmitter to send a codeword index rather than the error signal samples.

When applying CELP to multispectral imaging the following four different schemes are presented [15].

- **Multispectral Forward CELP** (MFCELP): The image is divided in macroblocks and blocks and a stable AR predictor for each macroblock is found. For each block the best excitation from a universal codebook is chosen.

- **Multispectral Segmented Forward CELP** (MSFCELP): The image is divided in texture regions using multispectral segmentation techniques and a stable AR predictor for each macroblock is found. Then depending on the region of the block different codebooks are used.

- **Multispectral Segmented CELP** (MSCELP): The image is divided in texture regions using multispectral segmentation techniques and a stable AR predictor for each region is found. Then depending on the region of the block different codebooks are used.

- **Multispectral Segmented Backward CELP** (MSBCELP): This method is identical to MSCELP but the AR coefficients are backward updated as opposed to forward updated.
2.3.5.6 SLSQ

*Spectral Oriented Least Squares* (SLSQ) is a low complexity algorithm for hyperspectral compression data (AVIRIS specifically) that employs linear prediction along with least squares optimization followed by entropy coding of the prediction error [19]. On this scheme the predictor is optimized for each pixel and for each band; given a reference plane and a tridimensional subset of past data an optimal linear predictor, in the least square sense, is determined for each sample.

Under SLSQ two prediction contexts are defined. As shown in Figure 2.18 the context that provides the shorter distance between predicted and real sample is used to represent the pixel. Rather than using a fixed predictor the coefficients that minimize the energy of the prediction error are calculated and sent as side information.

2.3.5.7 PLT

As indicated before ultraspectral sounder data exhibit a spectral correlation that is generally much stronger than the spatial correlation. *Prediction Based Lower Triangular Transform* (PLT) [38] is based on linear prediction but uses all linear predictors below a given order. If \( X = [x_1, x_2, x_3, ..., x_{nc}] \) is the original ultraspectral sounder data consisting of \( ns \) observations by \( nc \) channels then \( Y = [y_1, y_2, y_3, ..., y_{nc}] \) is the \( ns \times nc \) prediction errors and the prediction-based lower triangular transform \( P \) is of the form

\[
P = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
p_{1,0} & 1 & 0 & \cdots & 0 \\
p_{2,0} & p_{2,1} & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
p_{n_c-1,0} & p_{n_c-1,1} & p_{n_c-1,2} & \cdots & 1
\end{bmatrix}
\]

such that the prediction error \( Y \) can be computed by \( Y = PX \).

2.4 Proposed Approach

The focus of this dissertation is compression of AIRS data cubes that include more than 40MB of data spanned over a couple of thousand bands. Specifically a novel architecture that integrates preprocessing and compression stages and provides very efficient lossy compression is presented. Preprocessing involves a series of reversible techniques designed to convert an
ultraspectral AIRS data cube into a speech-like signal that exposes redundancy in such a way that the compression stage can remove it more efficiently. Specifically, preprocessing includes a sequence of reversible operations that do not provide compression but redistribute the input data to obtain increased redundancy in preparation for the following stage.

Three techniques are hereby integrated to accomplish this goal: band normalization, band ordering and image scanning. Band normalization reduces the dynamic range of spectral bands while preserving their integrity. Band ordering rearranges them to maximize the interband correlation. Image scanning partitions the spectral bands into spatial blocks (of variable size) that are subjected to specific scanning techniques in order to maximize the intraband correlation while serializing them to convert the tridimensional data into a speech-like signal. In general these procedures induce intraband periodicity of highly correlated samples transforming the output into a predictable signal that can be modeled as an AR process and subjected to linear prediction for which a valid filter order is obtained by analyzing the prediction gain response. This signal is therefore compressed by means of linear prediction in a similar to fashion to a speech signal. Details of these procedures are presented in the following chapters.
Chapter 3

Preprocessing Stage

This chapter presents the series of techniques, initially introduced in [12], to which AIRS images are subjected in order to preprocess them before the compression stage itself. As described in Section 1.3.3 preprocessing involves a sequence of reversible operations that in spite of not providing compression they redistribute the input data to better expose redundancy in preparation for the following stage.

3.1 AIRS Images

In chapter 1 it was mentioned that an ultraspectral AIRS image is comprised of 2107 spectral bands of data with each band containing 90x135 pixels. AIRS images obtained throughout a day are divided into 240 granules in such a way that a single image is captured every 6 minutes. For this dissertation the experimental input data, to be used for the comparison between the different compression schemes, comes from granules collected by NASA on March 2, 2004. It is important to mention that these images have been used as performance measure in other compression studies [16, 20].

Figures 3.1 through 3.5 show eight representative bands of the original AIRS images corresponding to granules 16, 82, 126, 151 and 193 respectively. Note that high correlation between bands is evident even for those bands that are not contiguous. It is clear that band 2000 is more correlated to band 1000 than to band 1700 although the latter is closer in frequency. The figures show each spectral band as a monochromatic image tuned to the specific wavelength corresponding to that band number.
Figure 3.1: March 3, 2004 Granule 16

Figure 3.2: March 3, 2004 Granule 82
Figure 3.3: March 3, 2004 Granule 126

Figure 3.4: March 3, 2004 Granule 151
3.2 The Speech Analogy

All of the linear prediction techniques reviewed in chapter 2 share the fundamental concept that a sample can be approximated by a linear combination of interband and intraband samples. For all of them intraband prediction refers to approximating the current sample with samples from the previous band only. Since ultraspectral images exhibit high correlation in the spectral dimension, using samples from other bands (other than the previous one) may help improve the compression rate. It is therefore possible to draw an analogy between spectral bands and pitch periods as ultraspectral image compression is a similar scenario to speech compression where a sample is approximated by a linear combination of past samples including those belonging to different pitch periods.

Linear prediction has been used in the field of speech compression for over 40 years and it forms an integral part of almost all modern day speech coders. It would be advantageous to find a way to convert ultraspectral bands into speech-like frame structures and use some of the techniques available for speech in the compression of ultraspectral images.

The simplest form of linear prediction applied to speech compression is *differential pulse code modulation* (DPCM) where the filter is of first order with one fixed coefficient known to both receiver and transmitter and only the prediction error is transmitted. Under *adaptive*
DPCM (ADPCM) the first order prediction filter is adaptive and the coefficients are dynamically calculated on a frame-per-frame basis and transmitted along with the prediction error. By using a higher order filter more redundant information can be removed from the prediction error and reduce its dynamic range improving compression. Note that ADPCM constitutes the base of 3D APCM, which is used in image compression and it was discussed previously in chapter 2. Under regular pulse excitation (RPE) [43] the filter order is ten and the error signal is downsampled before being quantized and transmitted. Grouping representative error signals for each speech frame to a dictionary leads to the code excited linear prediction (CELP) [44] approach. In this case a codebook with a large number of codewords is used for reference to generate the prediction error allowing the transmitter to send a codeword index rather than the error signal samples. The CELP concept is extended by vector sum excited linear prediction (VSELP) [45] that relies on a prediction error generated by multiple codebooks. Low delay CELP (LDCELP) [46] is an alternative approach where instead of a forward linear predictor a backward linear predictor is applied to shorter frames (low delay). LDCELP coefficients are calculated on already received samples (there is no need to transmit them) and only the codeword index is transmitted. Algebraic CELP (ACELP) [47] is an attempt to reduce the computational cost of the standard CELP coders by the use of simple algebra or mathematical rules to create the excitation code vectors. Another popular encoding technique is Mixed Excitation Linear Prediction (MELP) [48] where based on the nature of the speech (whether it is a voiced (vowels) or unvoiced (consonants) sound) a different excitation model is used.

In general speech signals show high correlation between samples belonging to different pitch periods [40]. This is specially the case for a voiced frame (i.e. frame generated by vowels) like the one shown by Figure 3.6.

It can be seen that for two pitch periods the interpitch correlation is usually larger than the intrapitch one. Although the pitch period changes within a frame and depends on specific characteristics of the speaker (i.e. typically between 5 and 10 ms) the correlation is fairly high for relative positions within the pitch period. This is evident by zooming in into the voiced frame as seen by Figure 3.7.

By exploiting the similarities due to the repetition of pitch periods and spectral bands, for the speech and AIRS cases respectively, it is possible to open AIRS to a series of techniques important to the development of signal compression. The preprocessing stage plays the role of inducing the intraband periodicity that makes the signal predictable by means of linear filtering the same way speech is.
Figure 3.6: Voiced frame

Figure 3.7: Voiced frame (zoom)
3.3 Preprocessing Steps

The following subsections describe the steps required to convert an ultraspectral AIRS image into a speech-like unidimensional signal that can be more efficiently processed by means of linear prediction during preprocessing.

3.3.1 Band Preprocessing

Band normalization transforms the dynamic range of the bands uniforming their sample distribution across different spectral bands. Band normalization can be integrated with an optional subband decomposition stage that can be used to concentrate the energy distribution in specific spatial regions preparing the data for more efficient compression.

3.3.1.1 Band Normalization

The dynamic range (in bits) of the spectral bands, defined as $d_i$, depends on the band number $i$ ($0 \leq i \leq B - 1 = 2106$) as indicated in Table 3.1 below. As part of the preprocessing stage each band is first normalized and each sample is converted from its original range to a value in the $[0, 1]$ interval. Specifically the mean and range of the samples in any given band, defined as $m_i$ and $l_i$ respectively, are encoded and transmitted to the decoder for future reconstruction.

<table>
<thead>
<tr>
<th>Band range</th>
<th>Dynamic Range [bit]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-695</td>
<td>12</td>
</tr>
<tr>
<td>696-1641</td>
<td>13</td>
</tr>
<tr>
<td>1642-2106</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 3.1: Dynamic range per band ($d_i$)

Typically the mean is encoded using as many bits as indicated by the dynamic range of the samples in the corresponding spectral band. The range can be encoded using one bit less than the bits required to encode the mean (worst-case scenario) because the range divides the overall dynamic range into two equal regions. Equation (3.1) defines $r_n$ as the additional rate increase measured in bit-per-pixel-band (bppb) due to band normalization.

$$r_n = \frac{1}{BMN} \sum_{i=0}^{B-1} 2d_i - 1$$

(3.1)
$B$ is the number of spectral bands in the image, $M$ and $N$ are the width and height of each band and the $d_i$ coefficients represent the samples’ dynamic range of each band as shown in Table 3.1. When replacing $B = 2107$, $M = 90$ and $N = 135$, to match the values corresponding to the images presented in Section 3.1, the equation (3.1) leads to $r_n = 0.002$ bppb.

![Table 3.2: Mean/Range entropy per granule](image)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Mean [bit]</th>
<th>Range [bit]</th>
<th>$r_n$ [bppb]</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>5.55</td>
<td>9.78</td>
<td>0.0012</td>
</tr>
<tr>
<td>82</td>
<td>5.55</td>
<td>9.74</td>
<td>0.0012</td>
</tr>
<tr>
<td>126</td>
<td>5.03</td>
<td>8.87</td>
<td>0.0011</td>
</tr>
<tr>
<td>151</td>
<td>5.00</td>
<td>8.16</td>
<td>0.0010</td>
</tr>
<tr>
<td>193</td>
<td>5.12</td>
<td>8.90</td>
<td>0.0011</td>
</tr>
</tbody>
</table>

Other more efficient encoding schemes are possible (i.e. Huffman, Golomb) however in order to determine whether it is worth using them it is important to find a theoretical bound for the entropy. Table 3.2 enumerates the bits required to encode the mean and range of each of the granules and it also indicates the corresponding rate increase $r_n$ due to the encoding. It is important noticing that the rate due to the entropy encoding is in the same order of magnitude as the rate due to the proposed mean and range encoding technique and therefore there is no significant rate improvement that justifies the implementation of a more advanced scheme. Because the rate increase of equation (3.1) is negligible when compared to other rate factors (mostly due to the compression stage) and the correlation of the mean and range for contiguous bands is small enough, it is not worth implementing a separate prediction scheme to encode them.

### 3.3.1.2 Subband Decomposition

While performing band normalization it is optional to apply subband decomposition to partition each spectral band into a group of frequency subbands each one representing the same spatial region corresponding to the whole spectral band. DWT decomposition of signals, as seen in Section 2.1.3.3, provides a multiresolution approach using a pyramidal filter structure of quadrature mirror filters (QMF) pairs. In multiresolution analysis, it can be proved that decomposition of signals using DWT can be expressed in terms of finite impulse response (FIR) filters.

The multiresolution decomposition approach is shown in Figure 3.8 for two levels of decomposition. The original signal $x_n$ is decomposed into three signals during preprocessing; a
high frequency (H) signal \(y_{0,n}\), a low-high frequency (LH) signal \(y_{1,n}\) and a low-low frequency signal (LL) \(y_{2,n}\). The decomposed signals are then recombined to reconstruct \(x'_n\) on the post-processing side. The two-dimensional extension of DWT is essential for the transformation of two dimensional signals, such as ultraspectral spectral bands. The simplest way for the two-dimensional implementation of DWT is to perform the unidimensional DWT row-wise to produce an intermediate result and then perform the same unidimensional DWT column-wise on this intermediate result to produce the final result as shown by Figure 3.9. Applying the procedure twice leads to a second level decomposition like the one shown in Figure 3.10. Note the spatial plane is divided into 7 regions: HH1, HL1, LH1, HH2, HL2, LH2 and LL2.

For perfect reconstruction QMF filters must comply with the constraint below:
Figure 3.11: Subband Decomposition: March 3, 2004 Granule 16

\[ H_0(z)G_0(z) + H_1(z)G_1(z) = H_0^2(z) - H_0^2(-z) \]

in such way that all filters are determined by \( H_0(z) \) [56]. The Haar filters defined as:

\[
\begin{align*}
H_0(z) &= 1 + z^{-1} \\
H_1(z) &= 1 - z^{-1} \\
G_0(z) &= 1 + z^{-1} \\
G_1(z) &= -1 + z^{-1}
\end{align*}
\]

provide perfect reconstruction and they are the filters used for subband decomposition of the spectral bands in this dissertation.

Figures 3.11 through 3.14 show the results of applying second level decomposition to the images in Section 3.1. Subband decomposition is applied after the spectral bands have their mean removed but before the range is normalized. The importance of this optional procedure is that the energy of the spectral bands is subjected to spatial redistribution in such a way that region LL2 contains most of the energy while region HH1 contains the least amount of energy. Note that the dynamic range of the high frequency regions in these
Figure 3.12: Subband Decomposition: March 3, 2004 Granule 82

Figure 3.13: Subband Decomposition: March 3, 2004 Granule 126
Figure 3.14: Subband Decomposition: March 3, 2004 Granule 193

Figure 3.15: Subband Decomposition: March 3, 2004 Granule 151
Figure 3.16: Subband Decomposition: March 3, 2004 Granule 16 (Histogram Equalization)

Figure 3.17: Subband Decomposition: March 3, 2004 Granule 82 (Histogram Equalization)
Figure 3.18: Subband Decomposition: March 3, 2004 Granule 126 (Histogram Equalization)

Figure 3.19: Subband Decomposition: March 3, 2004 Granule 151 (Histogram Equalization)
figures is small when compared to low frequency regions such that simple visual inspection doesn’t provide any appreciable detail or contrast. Figures 3.16 through 3.20 show the same spectral bands enhanced after applying histogram equalization[50] in order to show details in high frequency regions. Because the dynamic range of samples in each region depends on their energy content, the compression stage can take advantage of this fact by changing compression parameters (i.e. linear prediction filter order) based on the region of origin of the samples being compressed. In a worst-case scenario the compressor may decide whether to compress or not a specific low energy region in order to accomplish a specific rate-distortion goal.

### 3.3.2 Band Ordering

In order to convert spectral bands into speech-like structures the bands must be first re-ordered using the band ordering technique described in Section 2.2.1. Specifically equation (2.8) is used to measure the similarity between bands in such a way that they are arranged by their similarity starting by an arbitrary band. As indicated previously, for the images presented in Section 3.1, the values of $M$ and $N$ are given by the dimensions of each band and therefore $M = 90$ and $N = 135$. The value of $D$ not only rules the computation speed but also affects the precision of the calculation of equation (2.8). Since the best-case
scenario corresponds to \( D = 1 \) for any given value of \( D \) is possible to calculate the change of the relative positions of each of the bands against their positions obtained when \( D = 1 \). Mathematically a coefficient \( \rho(d) \) can be defined to account for the effect of \( D = d \) in the computation of the similarity as shown below:

\[
\rho(d) = \frac{B}{\sum_{i=1}^{B-1} |p_d(p_{d-1}(i)) - p_d(p_{d-1}(i-1))| + 1}
\]

where \( B = 2107 \) is the number of spectral bands and \( p_d(i) \) and its inverse \( p_d^{-1}(j) \) are described in the next paragraph.

The unordered bands are originally located in positions 0 through \( B - 1 \) so when band ordering is applied each one receives a new location. If the position of the original unordered band is \( i \) the corresponding new position is \( p_d(i) \) and similarly if the new position is \( j \) the position of the corresponding unordered band is \( p_d^{-1}(j) \). Because \( \rho(d) \) is inversely proportional to the \( L_1 \) norm distance between two contiguous bands (when \( D = 1 \)) its value becomes smaller when the distance is larger. The \(+1\) factor in the denominator of \( \rho(d) \) accounts for the fact that the first band is always ordered and acts as reference because \( p_d(0) = 0 \) for any \( D \).

Note that \( 1 \leq |p_d(p_{d-1}(i)) - p_d(p_{d-1}(i-1))| \leq (B - 1) \) and therefore \( \rho(d) \) is bounded as \( \frac{1}{B} \leq \rho(d) \leq 1 \). When the spectral bands are unordered \( p_u(i) \) is defined as the position of each band \( i \) and \( \rho_u \) is the corresponding \( \rho \) coefficient. Since \( p_u(i) = i \) for the worst-case scenario \( \rho_u \) becomes:

\[
\rho_u = \frac{B}{\sum_{i=1}^{B-1} |p_{d-1}(i) - p_{d-1}(i-1)| + 1} \geq \frac{1}{B} = 0.000474
\]

| Table 3.3: \( \rho(D) \) vs Granule |
|------------------|--------|--------|--------|--------|--------|
| \( D \)          | Granule | 16     | 82     | 126    | 151    | 193    |
| 1                | 1.000  | 1.000  | 1.000  | 1.000  | 1.000  |
| 2                | 0.303  | 0.382  | 0.297  | 0.258  | 0.309  |
| 3                | 0.093  | 0.102  | 0.087  | 0.082  | 0.091  |
| 4                | 0.030  | 0.025  | 0.022  | 0.035  | 0.035  |
| 8                | 0.019  | 0.019  | 0.020  | 0.024  | 0.024  |
| 16               | 0.012  | 0.015  | 0.013  | 0.020  | 0.025  |
| 32               | 0.014  | 0.008  | 0.011  | 0.013  | 0.012  |
| \( \rho_u \)     | 0.034  | 0.033  | 0.035  | 0.038  | 0.038  |

70
Table 3.3 shows $\rho(D)$ for each of the granules presented in Section 3.1. As indicated by $\rho(D)$ only for $D \leq 3$ it is convenient to apply band ordering to take advantage of the correlation between disjoint bands since for all other cases $\rho(D) < \rho_u$. This means that for $D \geq 4$ band ordering provides similarity coefficients that are not accurate enough to represent the correlation between bands. As a consequence, from this point on in this dissertation, whenever band ordering is applied a value of $D = 3$ is used.

The side information used to signal the decoder on how to arrange the bands is the list of all $B$ bands. Rather than encoding $B$ coefficients each one containing $\log_2 B$ bits of information it is possible to encode $\frac{B}{2}$ coefficients at $\log_2 B$ bits, $\frac{B}{4}$ of the remaining $\frac{B}{2}$ coefficients at $\log_2 \frac{B}{2}$ bits, $\frac{B}{8}$ of the remaining $\frac{B}{4}$ coefficients at $\log_2 \frac{B}{4}$ bits and so on until only one coefficient remains. Equation (3.2) defines $r_o$ as the additional rate increase due to band ordering.

$$r_o = \frac{1}{BMN} \sum_{i=1}^{\log_2 B} \frac{B}{2^i} \left( 1 + \log_2 B - i \right)$$  (3.2)

When replacing the values discussed above in the equation (3.2) the rate increase results $r_o = 0.0008264$ bppb.

### 3.3.3 Image Scanning

After performing the steps in subsections 3.3.1.1 and 3.3.2, an already normalized and band ordered image is segmented into a combination variable size spatial of blocks (i.e. $2 \times 2$, $5 \times 5$ and $10 \times 10$). Each block is bias adjusted by removing its $DC$ component and receives an index for identification. For example using $5 \times 5$ blocks, each band ends up with $18 \times 27 = 486$ blocks laid out as shown in Figure 3.21. Since the size of the spectral bands is $90 \times 135$ when utilizing blocks that are not multiple of 5 in any of the spatial directions (width or height) blocks of smaller size are required to complete the band mapping. For instance if an image is segmented into $2 \times 2$ blocks, a combination of $45 \times 67 = 3015$ $2 \times 2$ blocks and 45 $1 \times 2$ blocks is used.

In general the goal is to have blocks that are small enough to induce large sample correlation but big enough to minimize the weight of the mean encoding required by the bias adjustment. For blocks of width $W$ and height $V$, whose bias adjustment is encoded using $b$ bits, the rate increase measured in bit-per-pixel-band (bppb) is defined by equation (3.3) as $r_a$. The factor $\xi$, explained in the next paragraph, indicates whether any prediction is performed on the bias adjustment; if no prediction is applied $\xi = 1$. Note that the block
size is constant per band since, as it will be seen later on this dissertation, the goal is to preserve the interband periodicity when linearly scanning the spectral bands as part of the preprocessing stage.

\[ r_a = \begin{cases} 
\frac{b\xi}{WV} & \xi = 1 \\
1 + \frac{b\xi}{WV} & \xi < 1 
\end{cases} \]  

Due to the band normalization described in Section 3.3.1.1 the value of the samples is bounded in the \([0, 1]\) interval. Figures 3.22, 3.23 and 3.24 show the distribution of the bias adjustment for the different block sizes collected from the images presented in Section 3.1 cumulative to all the granules of origin. As mentioned before since AIRS samples show high sample correlation, both in the spectral and spatial dimensions, it is expected that bias adjustment values (which are derived from samples) also follow the same behavior. Figure 3.25 shows the bias adjustment for the first spatial block \((0,0)\) measured across all bands and Figure 3.26 shows the bias adjustment for the first spectral band 0 measured across all blocks (block size \(5 \times 5\)).

Because of these spatial and spectral dependencies it is possible to utilize a linear prediction scheme to encode the bias adjustment in such a way that the current value can
Figure 3.22: Distribution of Bias Adjustment (blocks size $2 \times 2$)

Figure 3.23: Distribution of Bias Adjustment (blocks size $5 \times 5$)
Figure 3.24: Distribution of Bias Adjustment (blocks size $10 \times 10$)

Figure 3.25: Bias Adjustment at Block $(0,0)$
be expressed as a linear combination of values in spatial blocks in the current and previous bands. In its simplest form the current bias adjustment can be approximated by the bias adjustment of the same spatial block in the previous band and only update its value when the difference between current and previous values are larger than a certain predefined threshold (a percentage of change relative to a reference sample). In that case equation (3.3) is affected by a coefficient $\xi < 1$ that improves the rate by reducing the number of bits to be encoded. Table 3.4 shows $\xi$ for different combinations of block size and threshold based on the experimental data obtained from the images in Section 3.1. Note that when $\xi < 1$ at least one bit per spatial block has to be encoded to order to indicate whether the bias adjustment value must be updated or not as part of the regular encoded bias adjustment information. When the acceptance threshold is larger, the rate improves because fewer bias adjustment values are encoded and when the block size increases the rate also improves (for the same threshold level) because the bias adjustment value includes more samples in its calculation and therefore provides a better estimation.

In order to minimize the complexity of the encoding scheme a fixed number of bits per bias adjustment value is worth considering; to be specific since the bias adjustment distribution can be considered uniform in the range between 0.3 and 0.9 (regardless of the
Table 3.4: Bias Adjustment ξ Coefficient

<table>
<thead>
<tr>
<th>Threshold</th>
<th>2 × 2</th>
<th>5 × 5</th>
<th>10 × 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 %</td>
<td>0.77</td>
<td>0.47</td>
<td>0.45</td>
</tr>
<tr>
<td>5 %</td>
<td>0.43</td>
<td>0.13</td>
<td>0.12</td>
</tr>
<tr>
<td>10 %</td>
<td>0.26</td>
<td>0.05</td>
<td>0.04</td>
</tr>
</tbody>
</table>

block size) an uniform quantizer can be used in a compression scenario. To guarantee complete reconstruction of the bias adjustment value the number of bits required to encode it, \( b \), must be equal or larger than the number of bits required to represent the range which was calculated during the band normalization presented in Section 3.3.1.1. Since the range depends on the band being considered, for a generic \( i^{th} \) band, the bias adjustment contribution is \( b_i \) defined by the equation below:

\[
b_i = \lceil \log_2 (l_i) \rceil
\]

Using the results obtained by applying band normalization \( b \) is generated from the average \( b_i \) taken for each band. In other words when \( b = \frac{1}{B} \sum_{i=0}^{B} b_i \) is combined with equation (3.3) the values presented in Table 3.5 are obtained. In average a 10-bit quantization of the bias adjustment value allows full reconstruction during the postprocessing stage.

Table 3.5: Bias Adjustment Rate

<table>
<thead>
<tr>
<th>Granule</th>
<th>2 × 2 [bppb]</th>
<th>5 × 5 [bppb]</th>
<th>10 × 10 [bppb]</th>
<th>b [bits]</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>2.25</td>
<td>0.36</td>
<td>0.09</td>
<td>9</td>
</tr>
<tr>
<td>82</td>
<td>2.25</td>
<td>0.36</td>
<td>0.09</td>
<td>9</td>
</tr>
<tr>
<td>126</td>
<td>2.50</td>
<td>0.40</td>
<td>0.10</td>
<td>10</td>
</tr>
<tr>
<td>151</td>
<td>2.50</td>
<td>0.40</td>
<td>0.10</td>
<td>10</td>
</tr>
<tr>
<td>193</td>
<td>2.50</td>
<td>0.40</td>
<td>0.10</td>
<td>10</td>
</tr>
</tbody>
</table>

Knowing the statistical distribution of the range, obtained during band normalization, it is possible to estimate \( b \) by calculating the expectation of the number of bits taken to represent the range. This together with the fact that \( \log_2(x) \) is a concave function provides the necessary conditions to apply Jensen’s inequality [42] leading to the relationship indicated by equation (3.4):

\[
b_i = \lceil E(\log_2 (l_i)) \rceil \leq \lceil \log_2 (E(l_i)) \rceil \quad 0 \leq i \leq B - 1 \quad (3.4)
\]
Figure 3.27: Band Range Distribution

Figure 3.27 shows the range distribution obtained from experimental data for all granules. Assuming a triangular distribution with base $b = 3000$ the expectation of the range is $E(x) = \frac{b}{3} = 1000$ which yields to $b \leq 10$ bits. This complies with the information shown in table 3.5.

The goal of the image scanning stage is to convert the partially preprocessed data into a speech-like unidimensional signal that can be compressed by means of linear prediction. A way to do this is by sampling blocks in the same spatial location of each spectral band in a specific fashion and then concatenating the sampled data into a single unidimensional signal as shown in the example of Figure 3.28. Since the idea is to minimize high variations to maximize the intrablock correlation during the compression stage, and therefore take advantage of the linear prediction, a method of scanning that visits pixels of the block in horizontal and vertical directions is needed. The best scanning technique is the one that once applied introduces the largest shift to low frequency components. It is important to mention that although there exist techniques that provides generic scanning for blocks of arbitrary sizes like peano [54] only a preset of scanning schemes are considered in this dissertation in order to provide efficiency. Figures 3.29 through 3.32 show 4 types of zig-zag scanning, Figures 3.33 and 3.34 show 2 types of progressive scanning and finally Figures 3.35 and 3.36 show 2 types of linear scanning.

Each block in the initial spectral band is evaluated for the best scanning scheme that
Figure 3.28: Band mapping

Figure 3.29: Zig-Zag Scanning (type 1)

Figure 3.30: Zig-Zag Scanning (type 2)

Figure 3.31: Zig-Zag Scanning (type 3)
Figure 3.32: Zig-Zag Scanning (type 4)

Figure 3.33: Progressive Scanning (type 1)

Figure 3.34: Progressive Scanning (type 2)

Figure 3.35: Linear Scanning (type 1)
provides the maximum intrablock correlation. Because interband correlation is high due to
the nature of ultraspectral images, the use of the initial band to compute the intrablock
correlation is sufficient. Equation (3.5) defines $r_s$ as the rate increase, measured in bit-per-
pixel-band (bppb), due to the encoding of this information.

$$r_s = \frac{t}{BWV}$$

(3.5)

$B$ is the number of bands, $W$ and $V$ are the width and height of each block and $t$ is the
number of bits required to encode the scanning scheme. Since there are 8 schemes $t = 3$ bits
and therefore for $2 \times 2$ blocks $r_s = 0.00035$ bppb, for $5 \times 5$ blocks $r_s = 0.000057$ bppb and
for $10 \times 10$ blocks $r_s = 0.000014$ bppb. Because these values are negligible when compared
to other rate factors there is no need to use entropy encoding due to the extra complexity
this would impose to the overall procedure.

In order to evaluate the efficiency of a given scanning technique it is necessary to check
the distribution of the autocorrelation of the scanned signal. Because the Wiener-Khinchin
theorem [52] relates the autocorrelation function to the power spectral density via the Fourier
Transform it is intuitive to think that the autocorrelation provides enough information to
measure the power shift of the scanned signal into low frequency after applying a specific
scheme. For any given scanned sequence $s_{k,n}$, obtained using the $k^{th}$ scanning technique
($k = 0 - 3$ (zig-zag), $k = 4 - 5$ (progressive) or $k = 6 - 7$ (linear)), equation (3.6) provides
a new coefficient $\delta_k$ that tells how correlated the samples in $s_{k,n}$ are. Note that $R_{k,i}$ is the
estimated $i^{th}$ autocorrelation coefficient of $s_{k,n}$ calculated as defined in Section 2.1.5. In the
worst-case scenario, for a signal $s_{k,n}$ that shows no sample correlation (such as white noise),
$\delta_k = 0$. In general, however, $0 \leq \delta_k < 1$ for all signals.

$$\delta_k = 1 - \frac{R_{k,0}^2}{\sum_{i=0}^{n} R_{k,i}^2}$$

(3.6)
Table 3.6: Block Scan Analysis (2 × 2 blocks)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Scan Type</th>
<th>Zig-Zag</th>
<th>Progressive</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td>1577</td>
<td>1438</td>
<td>3015</td>
</tr>
<tr>
<td>82</td>
<td></td>
<td>1592</td>
<td>1423</td>
<td>3015</td>
</tr>
<tr>
<td>126</td>
<td></td>
<td>1591</td>
<td>1424</td>
<td>3015</td>
</tr>
<tr>
<td>151</td>
<td></td>
<td>1566</td>
<td>1449</td>
<td>3015</td>
</tr>
<tr>
<td>193</td>
<td></td>
<td>1556</td>
<td>1459</td>
<td>3015</td>
</tr>
</tbody>
</table>

Table 3.7: Block Scan Analysis (5 × 5 blocks)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Scan Type</th>
<th>Zig-Zag</th>
<th>Progressive</th>
<th>Linear</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td>168</td>
<td>165</td>
<td>153</td>
<td>486</td>
</tr>
<tr>
<td>82</td>
<td></td>
<td>179</td>
<td>142</td>
<td>165</td>
<td>486</td>
</tr>
<tr>
<td>126</td>
<td></td>
<td>170</td>
<td>166</td>
<td>150</td>
<td>486</td>
</tr>
<tr>
<td>151</td>
<td></td>
<td>178</td>
<td>148</td>
<td>160</td>
<td>486</td>
</tr>
<tr>
<td>193</td>
<td></td>
<td>180</td>
<td>163</td>
<td>143</td>
<td>486</td>
</tr>
</tbody>
</table>

Table 3.8: Block Scan Analysis (10 × 10 blocks)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Scan Type</th>
<th>Zig-Zag</th>
<th>Progressive</th>
<th>Linear</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td>48</td>
<td>30</td>
<td>39</td>
<td>117</td>
</tr>
<tr>
<td>82</td>
<td></td>
<td>56</td>
<td>26</td>
<td>35</td>
<td>117</td>
</tr>
<tr>
<td>126</td>
<td></td>
<td>62</td>
<td>21</td>
<td>34</td>
<td>117</td>
</tr>
<tr>
<td>151</td>
<td></td>
<td>61</td>
<td>22</td>
<td>34</td>
<td>117</td>
</tr>
<tr>
<td>193</td>
<td></td>
<td>63</td>
<td>19</td>
<td>35</td>
<td>117</td>
</tr>
</tbody>
</table>

Tables 3.6 through 3.8 show the distribution of the best scanning techniques for each of granules presented in Section 3.1. The tables indicate how many spatial blocks are selected in each image for any of the scanning techniques (zig-zag, progressive or linear) based on the results obtained by computing $\max_k \delta_k$. Note that for blocks of size $2 \times 2$ zig-zag scanning produces the same output sequence as linear scanning and thus for the latter procedure results are not shown in Table 3.6.

Since all three techniques are, from the statistical point of view, almost uniformly distributed (probability tends to $1/2$ for $2 \times 2$ blocks and probability tends to $1/3$ for $5 \times 5$ and $10 \times 10$ blocks) entropy encoding, using $b = 3$ bits (for $5 \times 5$ and $10 \times 10$ blocks) in equation (3.5), guarantees the most efficient way to propagate this information to the decoder.
In spite of its almost uniform distribution it is zig-zag the scanning technique that repeats itself the most and therefore can be used by default for all blocks avoiding the rate increase generated by the encoding of a particular scanning method.

The last part of the preprocessing stage involves serializing the scanned blocks in order to convert the tridimensional data into a speech-like unidimensional signal as shown in Figure 3.28. The following steps summarize the complete conversion procedure given a normalized and band ordered input image.

i. Identify each spatial block with a number according to its location on a given spectral band (see figure 3.21)

ii. Identify each spectral band with a number according to its location (which is based on its correlation to neighboring bands)

iii. Let the set of output samples $O = \emptyset$ (empty set of samples)

iv. Let the block counter $c = 0$

v. Let the band counter $q = 0$

vi. Determine the scanning type (zig-zag, progressive or linear) for spatial block $c$ in spectral band $q$
Figure 3.38: Unidimensional AIRS Data (zoom) (2 × 2 blocks)

Figure 3.39: Unidimensional AIRS Data (5 × 5 blocks)
Figure 3.40: Unidimensional AIRS Data (zoom) (5 × 5 blocks)

Figure 3.41: Unidimensional AIRS Data (10 × 10 blocks)
vii. Scan spatial block \(c\) in band \(q\), concatenate the sequence of samples to \(O\), increment \(q\) and repeat (vii) while \(q < B\)

viii. Increment \(c\) and go to (v) while \(c < \frac{MN}{WV}\)

ix. \(O\) is a set of samples of a speech-like signal composed of rearranged samples of the input image

If the optional procedure of subband decomposition, described in Section 3.3.1.2, is applied, the output signal presents sections where the dynamic range of the samples is such that specific compression parameters can be modified to improve the overall compression ratio.

Figures 3.37, 3.39 and 3.41 show short sequences of a typical signal obtained by following steps (i) through (ix) applied to granule 82 when using \(2 \times 2\), \(5 \times 5\) and \(10 \times 10\) blocks respectively. Figures 3.38, 3.40 and 3.42 zoom in into fragments of such sequences for each of the block sizes mentioned above. It can be seen that for two signal periods the interband (interperiod) correlation is larger than the intraband (intraperiod) correlation. This is a similar scenario to the one described for speech processing in Section 3.2 but, as opposed to speech signals where the pitch or period is of variable length, in this case the period is fixed and identical to the scanning block area. A fixed period, as it will be seen later in this
dissertation, provides the advantage of eliminating its estimation and thus simplifying the compression stage.

3.4 Summary

This chapter introduced a series of reversible techniques destined to convert an ultraspectral AIRS image into a speech-like signal that presents redundancy in such a way that the compression stage removes it more efficiently. The spectral bands are first normalized by scaling each sample into the $[0, 1]$ interval while generating mean and range values as subproducts that are sent to the decoder for postprocessing. The resulting normalized image is fed to the band ordering stage which improves the correlation between contiguous spectral bands as it takes advantage of the fact that ultraspectral images exhibit strong correlation between disjoint spectral regions. The encoder then generates a bitstream that signals the decoder the specific spectral band sequence.

Bands are then segmented into spatial blocks of a particular size and have their bias adjusted by removing its mean component and encoding it as part of the preprocessing data. Blocks belonging to the same spatial location are then scanned using a specific scanning technique (zig-zag, progressive, linear) and concatenated to produce the unidimensional output signal. The scanning technique index is encoded and made available to the decoder.

The overall rate increase due to the preprocessing stage is obtained by combining equations (3.1), (3.2), (3.3) and (3.5) into the single equation (3.7) shown below.

$$r_p = \frac{1}{BMN} \left[ \sum_{i=0}^{B-1} 2d_i - 1 + \sum_{i=1}^{\log_2 B} \frac{B}{2^i} \left[ 1 + \log_2 B - i \right] \right] + \frac{1}{WV} \left[ b\xi + \frac{t}{B} \right]$$  \hspace{1cm} (3.7)

$B$ is the number of spectral bands in the image, $M$ and $N$ are the width and height of each band, $W$ and $V$ are the width and height of each block, $b$ is the number of bits required to encode the bias adjustment, $\xi$ signals the rate improvement due to bias adjustment prediction, $t$ is the number of bits required to encode the scanning scheme and $d_i$ is the dynamic range of each band where $i$ is between 0 and $B$.

Since for AIRS images $B \gg 1$ and $MN \gg 1$, $r_p \to \frac{k\xi}{WV}$ because it is dominated by the bias adjustment encoding term. As it was seen in Section 3.3.3 $b$ must be around 10 bits in average to provide full reversibility of the preprocessing stage for the images in Section 3.1 and $\xi = 1$ to indicate lack of bias adjustment prediction. This basically means that the final rate increase due to preprocessing is dependent almost exclusively in the width and height.
of the spatial blocks used to segment the spectral bands. As a gross approximation for $2 \times 2$ blocks $r_p \rightarrow 2.5$ bppb, for $5 \times 5$ blocks $r_p \rightarrow 0.4$ bppb and for $10 \times 10$ blocks $r_p \rightarrow 0.1$ bppb.

When implementing equation (3.7) to calculate the preprocessing rate, without incurring in any approximation, the following fixed parameters (that comply with the AIRS images of Section 3.1) are used: $B = 2107$, $M = 90$, $N = 135$, $b = 10$, $\xi = 1$, $t = 3$ and $d_i$ as defined in Table 3.1. The corresponding rates are $r_p \rightarrow 2.5032$ bppb for $2 \times 2$ blocks, $r_p \rightarrow 0.4029$ bppb for $5 \times 5$ blocks and $r_p \rightarrow 0.1029$ bppb for $10 \times 10$ blocks.

Figures 3.43 through 3.45 show the autocorrelation of the short sequences, previously shown in Figures 3.37 through 3.42, obtained by applying steps (i) through (ix) to granule 82 when using $2 \times 2$, $5 \times 5$ and $10 \times 10$ blocks respectively. They show, as expected, that the autocorrelation of the signals, regardless of the block size, decreases when the lag increases. Note that, also as expected, the interband (interperiod) correlation is larger than the intraband (intraperiod) correlation in a fashion that resembles speech signals.

Another point of contact to speech [40] is the fact that the signals follow a Laplacian distribution [41] with probability density function (pdf) $f(x) = \frac{\lambda}{2} e^{-\lambda |x|}$. Figures 3.46 through 3.48 show the actual sample distribution against the theoretical Laplacian distribution obtained when $\lambda = 150$, $\lambda = 115$ and $\lambda = 65$ for $2 \times 2$ blocks, $5 \times 5$ blocks and $10 \times 10$ blocks respec-
Figure 3.44: Autocorrelation Output ($5 \times 5$ blocks)

Figure 3.45: Autocorrelation Output ($10 \times 10$ blocks)
Figure 3.46: Real Data vs Laplacian $\lambda = 150$ ($2 \times 2$ blocks)

tively. Note that the variance of the Laplacian distribution is given by $\sigma^2 = \frac{2}{\lambda^2}$ and decreases when $\lambda$ increases. Because spatial blocks of bigger area tend to include more samples, the spread of their sample distribution is wider and therefore can be modeled using distributions with larger values of $\lambda$.

The nature of the preprocessing stage guarantees that the transformation is fully reversible, introduces no distortion and prepares the speech-like output signal for the compression stage. The next chapter will present this new stage as it will describe the non-reversible procedures in charge of removing redundancy and achieving compression while introducing distortion in a trade-off that defines a rate-distortion scenario.
Figure 3.47: Real Data vs Laplacian $\lambda = 115$ (5 $\times$ 5 blocks)

Figure 3.48: Real Data vs Laplacian $\lambda = 65$ (10 $\times$ 10 blocks)
Chapter 4

Compression Stage

Chapter 3 presented a sequence of steps to convert an ultraspectral image into a speech-like signal that can be more efficiently compressed. This chapter deals with the techniques to which those signals are subjected in order to perform compression. Specifically the advantages of linear prediction based compression and its performance are detailed. Note that the research presented in this chapter has been initially introduced in [20].

4.1 Prediction Gain

Since linear-prediction is an integral part of almost all modern day speech coding algorithms, a similar approach can be applied to the speech-like signal obtained during the preprocessing stage. For the most general case the basic assumption is that the converted sequence can be modeled as an AR signal and that linear prediction allows redundancy removal where information repeated in an event is eliminated. Under the AR model the current sample is predicted as a combination of $L$ past samples as detailed in Section 2.1.5. Note that to perform this calculation the input signal is first partitioned into frames, procedure to be detailed later in Section 4.4, that are sequentially subjected to linear prediction. The by-product of this process is a concatenation of prediction error frames that compose the error signal.

One way to measure the compression performance is by means of the prediction gain [40] which is defined as $G$ by equation (4.1).

$$ G = 10 \log_{10} \left( \frac{E\{s_n^2\}}{E\{e_n^2\}} \right) $$

(4.1)
It is clear that as the prediction error $e_n$ (presented in equation (2.3)) decreases the prediction gain grows regardless of the $s_n$ signal value. This dependency of the prediction gain on the prediction error, makes the prediction gain a function of the prediction order (described in equation (2.2)). A larger prediction order improves the approximation of the estimated signal $\hat{s}_n$ and as a consequence the prediction error results smaller. It is therefore expected to find $G(L_1) \leq G(L_2)$ if $L_1 \leq L_2$ although an excessively high order can lead to overfitting in such a way that undesirable errors are introduced. Ideally under forward prediction the filter order needs to be minimized in order to improve the compression ratio since fewer filter coefficients need to be transmitted.

For each of the granules presented in Section 3.1, Figures 4.1, 4.2 and 4.3 show the prediction gain as a function of the filter order for scanning block sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively. The shape of the plots is independent of the granule in such a way that there exists a threshold that determines whether the prediction gain grows or stays constant when the prediction order is below or above it.

Specifically the prediction gain grows fast when $L \leq 4$, $L \leq 25$ and $L \leq 100$ for block sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively so the inflection point is equal to the area of the corresponding scanning block. Further increasing the prediction order doesn’t produce significant gain improvements. If $L$ is below this threshold the number of LP coefficients is not enough to remove the correlation between samples one period apart. For $L$ above
Figure 4.2: Prediction Gain (5 × 5 blocks)

Figure 4.3: Prediction Gain (10 × 10 blocks)
the threshold, however, the linear predictor is capable of modeling the correlation between samples one period apart, therefore leading to a substantial improvement in prediction gain.

Another way to see the filter order effect on the prediction error is shown by Figures 4.4, 4.5 and 4.6 for block sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively. Each figure compares a small segment of the prediction error obtained for a given filter order against the same segment when no prediction is performed. It is evident that when the prediction order increases the energy content of the prediction error decreases as signal redundancy is removed.

Similarly Figures 4.7, 4.8 and 4.9 show the autocorrelation of the prediction error for block sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively. It can be seen that the figures show peaks at delay positions multiple of the area of the scanning block. When the filter order is equal to the area of the scanning block the redundancy induced by the prediction error periodicity is removed. Note that even in this extreme case there are still smaller peaks at locations multiple of the block area. This implies the dependency between samples more than a period away. In general, as the filter order increases and predictability is removed, the prediction error tends to behave more like white noise.

### 4.2 Long term prediction

As seen in Section 2.1.5 the prediction error tends to be approximately a white noise signal that can be further whitened when removing its correlation peaks without significantly increasing the filter order (increasing complexity and coefficients to be transmitted) by cascading long term prediction filters [59] to the short term prediction filter as shown in Figure 4.10.
Figure 4.5: Prediction Error Comparison (5 × 5 blocks)
Figure 4.6: Prediction Error Comparison (10 × 10 blocks)
Figure 4.7: Prediction Error Autocorrelation (2 × 2 blocks)
Figure 4.8: Prediction Error Autocorrelation (5 × 5 blocks)
Figure 4.9: Prediction Error Autocorrelation (10 × 10 blocks)

Figure 4.10: Filter Cascade
In general for a single long term predictor the prediction error is calculated based on the prediction error a number $k$ of periods away of the current sample as seen by equation (4.2).

$$\hat{e}_n = -be_{n-kT}$$  \hspace{1cm} (4.2)

In this case $T$ represents the inter-band period (equal to the area of the scanning block $T = WV$). The mean-squared error of this long term prediction is as indicated by equation (4.3).

$$J = \sum_n (e_n + be_{n-kT})^2$$  \hspace{1cm} (4.3)

The optimal value of $b$, obtained by minimizing the mean-squared prediction error, is given by the expression (4.4).

$$b = -\frac{\sum n e_n e_{n-kT}}{\sum n e_{n-kT}^2}$$  \hspace{1cm} (4.4)

When cascading multiple stages of long term filters each stage corresponds a specific value of $k$. Specifically first, second and third stages are processed when $k = 2, k = 3$ and $k = 4$ respectively. Figures 4.11 through 4.13 show the autocorrelation of the prediction error when applying a different number of stages for scanning block sizes $2 \times 2, 5 \times 5$ and $10 \times 10$. As expected each additional stage adds an additional LP coefficient to the list of LP coefficients required to synthesize the ultraspectral data in the decoder.

The long term prediction gain obtained when using three stages is very close to the prediction gain obtained when using a short term prediction filter of equivalent order (four times the scanning block area). Table 4.1, 4.2 and 4.3 show the prediction gain obtained when using different long term prediction stages as well as a short term prediction (of quadruple block size order) as a function of the granule for block sizes $2 \times 2, 5 \times 5$ and $10 \times 10$ respectively. These results indicate that the long term filters account for reducing most of the prediction error induced by short term filters of lower order.

Summarizing, the use of long term prediction improves compression while providing a prediction gain of the same order of magnitude as the one generated by a short term predictor of similar filter order. The relative compression ratio, defined as $k_L$, due to long term prediction can be computed by means of the expression (4.5) below.

$$k_L = \frac{WV (n + 1)}{WV + n} \hspace{1cm} (4.5)$$
Figure 4.11: Long Term Prediction Error Autocorrelation (2 × 2 blocks)
Figure 4.12: Long Term Prediction Error Autocorrelation (5 × 5 blocks)
Figure 4.13: Long Term Prediction Error Autocorrelation (10 × 10 blocks)
Table 4.1: Long Term Prediction Gain vs Granule (2 × 2 blocks)

<table>
<thead>
<tr>
<th>Granule</th>
<th>No Stages</th>
<th>1 Stage</th>
<th>2 Stages</th>
<th>3 Stages</th>
<th>Order 16</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>5.22</td>
<td>5.25</td>
<td>5.29</td>
<td>5.32</td>
<td>5.39</td>
</tr>
<tr>
<td>82</td>
<td>4.81</td>
<td>4.85</td>
<td>4.92</td>
<td>4.98</td>
<td>5.02</td>
</tr>
<tr>
<td>126</td>
<td>6.44</td>
<td>6.46</td>
<td>6.49</td>
<td>6.52</td>
<td>6.55</td>
</tr>
<tr>
<td>151</td>
<td>6.20</td>
<td>6.24</td>
<td>6.31</td>
<td>6.34</td>
<td>6.62</td>
</tr>
<tr>
<td>193</td>
<td>6.97</td>
<td>6.99</td>
<td>7.05</td>
<td>7.09</td>
<td>7.11</td>
</tr>
</tbody>
</table>

Table 4.2: Long Term Prediction Gain vs Granule (5 × 5 blocks)

<table>
<thead>
<tr>
<th>Granule</th>
<th>No Stages</th>
<th>1 Stage</th>
<th>2 Stages</th>
<th>3 Stages</th>
<th>Order 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>7.30</td>
<td>7.41</td>
<td>7.56</td>
<td>7.69</td>
<td>7.71</td>
</tr>
<tr>
<td>82</td>
<td>6.40</td>
<td>6.45</td>
<td>6.65</td>
<td>6.73</td>
<td>6.81</td>
</tr>
<tr>
<td>126</td>
<td>9.02</td>
<td>9.09</td>
<td>9.12</td>
<td>9.05</td>
<td>9.21</td>
</tr>
<tr>
<td>151</td>
<td>7.49</td>
<td>7.56</td>
<td>7.62</td>
<td>7.72</td>
<td>7.88</td>
</tr>
</tbody>
</table>

where \( n \) is the number of long term stages to be considered. Table 4.4 shows the compression ratio as a function of the scanning block size and the number of long term prediction stages. Equation (4.5) is bound by \( k_L \rightarrow WV \) as \( n \rightarrow \infty \) in such a way that relative compression due to long term prediction is never larger than the scanning block area.

Note that the overall results presented by Tables 4.1 through 4.3 indicate that there is no significant improvement by using long term prediction alongside standard short term prediction. The most important performance improvement occurs when utilizing short term prediction of filter order equal to the scanning block area as prediction gain is maximized while maintaining the overall compression ratio low.

### 4.3 Input Frame Size

Typically the input signal frame size must be at least as large as the filter order and, depending how much past information is to be included in the LP coefficient calculation, it can be a multiple of it. Increasing the frame size smooths the LP coefficient computation as it removes any transient behavior that may affect the dynamic range, therefore improving the compression. Larger input frames, however, incorporate older samples into the LP coefficient computation that introduce distortion to the prediction error frames increasing their energy content and forcing LP coefficient recalculation. Each LP coefficient recalculation decreases the compression rate as additional coefficients are encoded. As indicated in Section 4.2 the
optimal linear filter order occurs when $L = WV$ so the input frame size is $f = wL = wWV$ where $w$ is an integer such that $1 \leq w \leq k$.

Figures 4.14, 4.15 and 4.16 show the prediction gain $G$, defined by equation (4.1), as a function of the frame size $f$ for scanning blocks sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively. As it can be seen in the plots the maximum prediction gain occurs at $f = 32$ for $2 \times 2$ blocks, at $f = 400$ for $5 \times 5$ blocks and at $f = 800$ for $10 \times 10$ blocks.

### 4.4 Prediction Coefficients

The LP coefficients of the linear prediction can be obtained by solving the normal equation seen in Chapter 2.1.5. Specifically the matrix inverse $a = -R^{-1}r$ contains the linear prediction coefficients to be used to remove the redundancy in the input signal. In speech scenarios, the LP coefficients are calculated on a per input frame basis (typically every frame contains 10 ms of speech), encoded and transmitted along with the resulting error signal. In the compression architecture presented in this dissertation the LP coefficients are calculated periodically based on the energy of the prediction error and how it changes from frame to frame. Initially a set of default LP coefficients are assumed and, for any given input frame, the prediction error is computed in such way that if the relative energy of the error, *error gain*, is larger than a predefined threshold LP coefficients are recalculated and they become the default set.
Figure 4.14: Prediction Gain vs Frame Size (2 $\times$ 2 blocks)

Figure 4.15: Prediction Gain vs Frame Size (5 $\times$ 5 blocks)
Specifically for a fixed input frame size $f = q \cdot W \cdot V$, the error gain $G_e$ is calculated using equation (4.6) on a per frame basis and compared against the aforementioned threshold to determine whether or not to compute LP coefficients. Figures 4.17, 4.18 and 4.19 plot the error threshold (in dB) versus the proportion of the total frames, defined as $\psi$, for which LP coefficient calculation (using the Durbin-Levinson algorithm detailed in Appendix B.2) is performed using granules of the sample dataset for $2 \times 2$, $5 \times 5$ and $10 \times 10$ scanning block sizes respectively. A 3.5 dB threshold guarantees that no more than $\frac{1}{3}$ of the input frames account for LP coefficient calculation.

There are efficient algorithms which take advantage of the special structure of the correlation matrix. One of them is the Levinson-Durbin algorithm [55] which provides reflection coefficients (RC) obtained through the LP coefficient. RC can be used for verification of the minimum phase property of the resultant prediction error filter. A system is minimum phase when its poles and zeros are inside the unit circle and therefore has a casual and stable inverse.

One of the issues with LP coefficients is that they possess a large dynamic range and a bound on their values cannot be found on a theoretical basis. This is not important if the
Figure 4.17: LP Coefficient Calculation per Frame vs Error Threshold (2 × 2 blocks)

Figure 4.18: LP Coefficient Calculation per Frame vs Error Threshold (5 × 5 blocks)
algorithms are implemented in a floating-point environment but are critical in fixed-point ones. The Leroux-Gueguen solution is a method to compute the RC from the autocorrelation values without dealing directly with the LP coefficients in such a way that the problems related to dynamic range in a fixed-point environment are eliminated [58].

RC represent an alternative to LP coefficients and there is a one-to-one correspondence between the two sets of parameters, in practical applications the log area ratios (LAR) are used to represent the RC for transmission. Another way to represent LP coefficients is by means of line spectral pairs (LSP) [61] which due to many desired properties (among them smaller sensitivity to quantization noise) they have received widespread acceptance in speech coding applications.

In the application of the linear prediction the resultant synthesis filter might become marginally stable because of the poles located too close to the unit circle. The problem is aggravated in fixed-point implementations where a marginally stable filter can actually become unstable after quantization and loss of precision. Stability can be improved by modifying the LP coefficients according to $a_i' = \gamma^i a_i$ with $\gamma < 1$ a positive constant. The operation moves all the poles of the filter radially toward the origin leading to improved stability [62]. By doing so, the original spectrum is bandwidth expanded and it becomes flatter specially around the peaks.

It is well known that most of the speech coders utilize some sort of VQ for the encoding
of the LP coefficients [53]. This is because of the superior performance when compared with scalar quantization. Practical deployment of VQ is limited mainly by its high complexity and many structured schemes are often employed. In general most methods deal exclusively with LSP vectors that exhibit a great deal of redundancy that can be explored more efficiently through VQ. This redundancy has to do with the dependency between elements of a frame (intraband) as well as elements from different frames (interband). Note that under speech compression the usage of Split VQ and MSVQ (both described in chapter 2) has been widely implemented.

It is possible to measure the distortion introduced by the quantization of the LP coefficients by calculating the spectral distortion ($S_D$). In general given two sets of LP coefficients $a_{1,i}$ and $a_{2,i}$ two AR processes are defined by expressions (4.7) and (4.8) respectively.

$$A_1(z) = 1 + \sum_{i=1}^{L} a_{1,i} z^{-i} \quad (4.7)$$

$$A_2(z) = 1 + \sum_{i=1}^{L} a_{2,i} z^{-i} \quad (4.8)$$

The power spectral densities (PSD) of these processes are given by equations 4.9 and 4.10 respectively.

$$S_1(e^{j\omega}) = \frac{1}{|A_1(e^{j\omega})|^2} \quad (4.9)$$

$$S_2(e^{j\omega}) = \frac{1}{|A_2(e^{j\omega})|^2} \quad (4.10)$$

The spectral distortion between $S_1$ and $S_2$ is defined by equation (4.11).

$$S_D^2 = \frac{1}{2\pi} \int_0^{2\pi} \left[ 10 \log_{10} \left( \frac{|A_2(e^{j\omega})|^2}{|A_1(e^{j\omega})|^2} \right) \right]^2 d\omega \quad (4.11)$$

The result is given in decibels and can be approximated in practice by sampling the PSD using $N$ points as seen by equation (4.12).

$$S_D^2 = \frac{1}{N} \sum_{n=0}^{N} \left[ 10 \log_{10} \left( \frac{|A_2(e^{j2\pi n/N})|^2}{|A_1(e^{j2\pi n/N})|^2} \right) \right]^2 \quad (4.12)$$
For speech coders transparent quantization is defined for those cases where the LP coefficients quantization does not introduce any perceptible distortion. It has been found an average SD of 1 dB is the satisfactory limit to establish transparent quantization [24].

When the LP coefficients are quantized using scalar quantization a number of bits, defined as $q_v$, are used to quantize each LP coefficient. Experimental data obtained for all granules and scanning block sizes show that the LP coefficients follow a Laplacian distribution as seen by Figure 4.20. The coefficients are uniformly quantized but logarithmic companding is applied in the encoder before quantization to take advantage of the distribution of the LP coefficients. Figures 4.21, 4.22 and 4.23 show the average spectral distortion obtained if the LP coefficients are scalar quantized when scanning blocks of size $2 \times 2$, $5 \times 5$ and $10 \times 10$ are used respectively. A value of $q_v = 7$ provides a spectral distortion $S_D$ close to 1 dB that can considered as an acceptable limit for transparent quantization.

$$r_l = \frac{\psi q_v WV}{f}$$  \hspace{1cm} (4.13)

Equation (4.13) defines $r_l$ as the additional rate increase measured in bit-per-pixel-band (bppb) due to LP coefficient encoding. This value is based on the assumption that short term prediction involves a filter order $L = WV$ as indicated in Section 4.2. When replacing $f = 400$, $\psi = \frac{1}{3}$ and $q_v = 7$, to match the optimal values corresponding to the experimental data
Figure 4.21: LP Coefficients Quantization (2 x 2 blocks)

Figure 4.22: LP Coefficients Quantization (5 x 5 blocks)
presented throughout this chapter, equation (4.13) leads to $r_1 = 0.2917$ bppb for scanning blocks of size $5 \times 5$.

### 4.5 Prediction Error

The prediction error signal is always quantized and transmitted in a process in which distortion is introduced. Prediction error encoding of speech signals has been a popular area of research for decades as it plays a central role in the recovery of the original signal. Typically and depending on the filter order the prediction error signal tends to be approximately a white noise signal, because all the redundancy has been removed, and only the pure random component is present [40].

The main question is how to encode the prediction error to minimize distortion and maximize the compression rate simultaneously. CELP [44] and its many variations VSELP [45], LDCELP [46] and ACELP [47], just to name a few, are the preferred techniques for encryption of the prediction error under speech compression. In fact as mentioned in chapter 2 the basic architecture of CELP has been extended for compression of multispectral combined with segmentation techniques in regular bidimensional prediction (where the reference is the previous band). It is possible to extend the speech coding principles of CELP to ultraspec-
Figure 4.24 shows the block diagram of the production model where in such a way that excitation sequence is extracted from the codebook through an index. The extracted excitation is scaled to the appropriate level and filtered by the cascade connection of the long and short term synthesis filters to yield the reconstructed data. For speech signal prediction, where the delay due to the short term filter order is much smaller than the pitch periodicity, the long term filter creates the periodicity in the signal associated with the fundamental period frequency and the short term filter generates the spectral envelope. For AIRS prediction, as indicated in section 4.2, the fundamental period frequency is induced by the short term filter order, which is equal to the scanning block area. In this case the long term filter doesn’t produce significant prediction gain improvements and it can be removed from the cascade of synthesis filters in Figure 4.24 (no associated long term parameters are needed to reconstruct the signal).

The codebook can be fixed or adaptive and can contain deterministic pulses or random noise. For simplicity in most cases the codebook is fixed and contains white noise samples such that the excitation index selects a white noise sequence from the codebook as input to the cascade connection of the synthesis filters. The whole AIRS data production model that the CELP coder relies on consists simply of a white noise source stimulating the synthesis filter.

Linear prediction systems can be classified into two types:
• *open-loop* when the compression parameters (i.e. prediction error signal, LP coefficients) are extracted from the input signal and quantized so they are later used for synthesis.

• *closed-loop* when the compression parameters (i.e. gain, LP coefficients) are used to synthesize the signal during encoding and fine-tuned as to generate the most accurate reconstruction.

CELP encoders use the closed-loop optimization procedure where the goal is to choose the best parameters so as to match as much as possible the synthetic signal with the original one. Figure 4.25 shows the block diagram of an encoder with the closed-loop approach. Since the signal is synthesized during encoding for analysis purposes, the principle is known as *analysis-by-synthesis* (AbS) [49]. Figure 4.26 shows the AbS encoder when the unidimensional input AIRS data is processed to generate the output bitstream composed of the codebook index, gain and the LP coefficients. The preprocessed input stream is segmented intro frames that are fed into the analysis filter in order to generate the LP coefficients. The LP coefficients
and the excitation codebook of size $U$ are the input to the synthesis filter in such a way that the reconstructed frame is compared against the original frame in order to determine the gain and the excitation codeword that minimizes the error. As seen in Section 4.3 the value of the frame size $f$ is proportional to the filter order $L$ as $f = wL$ where $w$ is an integer such that $1 \leq w \leq k$ in such a way that the maximum prediction gain occurs at $f = 32$ for $2 \times 2$ blocks, at $f = 400$ for $5 \times 5$ blocks and at $f = 800$ for $10 \times 10$ blocks. For each input frame a given codebook index and gain value are selected by the AbS algorithm and encoded using $\log_2 U$ and $q_g$ bits per frame respectively. Since the input frame has a Laplacian distribution it can be assumed that the gain will also follow a Laplacian distribution that can be encoded using a linear quantizer prior to logarithmic companding. Equation (4.14) defines $r_e$ as the additional rate increase due to prediction error encoding when closed-loop CELP is used.

$$r_e = \frac{\log_2 U + q_g}{f} \quad (4.14)$$

Figures 4.27, 4.28 and 4.29 show the distortion measured as PSNR as a function of the excitation codebook size for scanning blocks sizes $2 \times 2$, $5 \times 5$ and $10 \times 10$ respectively (with fixed $f$ and $q_g$). As described in Section 1.3.2 PSNR is calculated as $\text{PSNR} = 10 \log_{10} \left( \frac{\text{MAX}_i^2}{\text{MSE}} \right)$ where $\text{MAX}_i$ is the maximum possible pixel value and it $\text{MSE}$ is the mean square error. For any codebook of size larger than $U = 16384$ the distortion reduction is not significant.
Figure 4.28: Distortion vs Codebook Size (5 × 5 blocks)

Figure 4.29: Distortion vs Codebook Size (10 × 10 blocks)
enough to justify the additional complexity increase and performance degradation due to its implementation. Typically for $5 \times 5$ scanning blocks, and based on the experimental data, the most efficient way to encode the prediction error using a closed-loop CELP scheme consists of segmenting the input signal into frames of size $f = 400$ and using a codebook of size $U = 16384$ as excitation achieving an estimated rate increase due to error encoding equal to $r_e = 0.0525$ bppb.

As an alternative to closed-loop CELP based encoding it is possible to apply vector quantization to the prediction error by following an open-loop approach. To be specific, the prediction error is partitioned into frames of $\upsilon$ samples and vector quantized using the LBG algorithm detailed in Appendix B.1. The generated codebook, of size $U$, is transmitted to the decoder for use in the reconstruction of the prediction error. As seen in Figure 4.30 the prediction error samples follow a Laplacian distribution of parameter $\lambda = 40$ that can be companded using a logarithmic rule and then uniformly quantized. Extending the concept seen in the previous section, $q_\upsilon$ bits are used to quantize each sample of the codebook. Equation (4.15) defines $r_e$ as the additional rate increase due to prediction error encoding when open-loop vector quantization is used.

$$r_e = \frac{\log_2 U}{\upsilon} + \frac{U \upsilon q_\upsilon}{BMN}$$  \hspace{1cm} (4.15)
The first term takes into account the bits per sample due to the vector quantization and the second term accounts for the encoding of the codebook itself.

Table 4.5: Codeword Size \((v)\) vs Codebook Size \((U)\)

<table>
<thead>
<tr>
<th>Codebook Size ((U))</th>
<th>Codeword Size ((v))</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>89.0039</td>
</tr>
<tr>
<td>512</td>
<td>66.7529</td>
</tr>
<tr>
<td>1024</td>
<td>49.7547</td>
</tr>
<tr>
<td>2048</td>
<td>36.8991</td>
</tr>
<tr>
<td>4096</td>
<td>27.2518</td>
</tr>
</tbody>
</table>

In general the overhead due to transmitting the codebook is negligible when compared to the information used to encode the codewords as long as the frame size \(v\) is kept small enough. This is a desirable fact as computation restrictions and efficiency impose the use of small codewords. The value of \(v\) can be obtained by assuming that the codebook transmission rate can never be larger than an order of magnitude of the codeword encoding rate leading to \(v < \sqrt{\frac{BMN \log_2 U}{10Uq_v}}\). Typically for \(B = 2107, M = 90, N = 135\) and \(q_v = 7\) the value of \(v\) (codeword size) is shown as a function of \(U\) (codebook size) in Table 4.5. A codeword size of 32 samples is small enough to provide efficient vector quantization while guaranteeing minimal effect on the rate increase due to codebook encoding.
Figure 4.32: Distortion vs Codebook Size (5 \times 5 blocks)

Figure 4.33: Distortion vs Codebook Size (10 \times 10 blocks)
Figures 4.31, 4.32 and 4.33 show the distortion measured as PSNR as a function of
the codebook size \((U)\) when using \(2 \times 2\), \(5 \times 5\) and \(10 \times 10\) scanning blocks respectively.
Although the distortion decreases as the codebook size increases the most relevant increment
occurs when the codebook size is 2048 (11-bit codewords). Summarizing, based on the
experimental data, the most efficient way to quantize the prediction error using an open-
loop scheme consists of segmenting it into 32-sample frames and using a 2048-codeword
codebook to quantize them in which case the estimated rate increase due to error encoding
results \(r_c = 0.3617\) bppb.

4.6 Summary

This chapter introduced the techniques to which preprocessed images are subjected in order
to accomplish data compression. Specifically the signal is fed into a linear-prediction scheme
from which an error signal and linear prediction coefficients are obtained. It was discussed
that the best results, based on experimental data, result when short term prediction of order
\(L = WV\) is used and applied to an input frame 8 times larger than the filter order itself. The
LP coefficients obtained are scalar quantized (7-bit samples) using logarithmic companding
and only encoded when the resulting error changes based on a specific error gain of 3.5
dB. When implementing the closed-loop CELP approach the error signal is fragmented into
100-sample frames and using a 16384 excitation codeword codebook indices and gains are
sent the decoder for reconstruction. The overall rate increase due to the compression stage
is obtained by combining equations (4.13) and (4.14) into the single equation (4.6) shown
below.

\[
\frac{1}{f} = \psi q_v WV + \log_2 U + q_g
\]  

When replacing \(\psi = \frac{1}{3}, q_v = 7, f = 400, W = 5, V = 5, U = 16384\) and \(q_g = 7\) in
equation (4.6) the compression rate results \(r_c = 0.1983\) bppb.

When implementing the open-loop approach the error signal is segmented into 32-sample
vectors and using a 2048-codeword codebook they are encoded and transmitted to the de-
coder. On this circumstances the overall rate increase due to the compression stage is
obtained by combining equations (4.13) and (4.15) into the single equation (4.7) shown
below.

\[
\frac{1}{f} = \psi q_v WV + \frac{\log_2 U}{v} + \frac{U v q_v}{BMN}
\]  

121
When replacing \( U = 2048, \psi = \frac{1}{3}, v = 32, q_v = 7, f = 400, B = 2107, W = 5, V = 5, M = 90 \) and \( N = 135 \) in equation (4.7) the compression rate results \( r_c = 0.5075 \) bppb.

Note that \( r_c \) accounts for the rate due to the compression stage itself but doesn’t take into account the rate increase due to preprocessing. When combining both, preprocessing and compression, the total compression rate is given by expression (4.18).

\[
r_t = r_p + r_c
\]  

where \( r_c \) was previously defined in Section 3.4. Under these assumptions, for scanning blocks of size \( 5 \times 5 \) in the preprocessing stage, \( r_p = 0.4029 \) bppb and the estimated compound compression rate is \( r_t = 0.6012 \) bppb for CELP based encoding and \( r_t = 0.9104 \) bppb for vector quantization based encoding.

Table 4.6: CELP Compression Rate \( (\hat{r}_t) \) vs Granule

<table>
<thead>
<tr>
<th>Granule</th>
<th>( \hat{r}_t ) (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0.60</td>
<td>49.36</td>
<td>9.19</td>
<td>0.89</td>
</tr>
<tr>
<td>82</td>
<td>0.62</td>
<td>48.41</td>
<td>8.82</td>
<td>0.87</td>
</tr>
<tr>
<td>126</td>
<td>0.63</td>
<td>51.74</td>
<td>9.05</td>
<td>0.90</td>
</tr>
<tr>
<td>151</td>
<td>0.66</td>
<td>52.20</td>
<td>8.27</td>
<td>0.91</td>
</tr>
<tr>
<td>193</td>
<td>0.62</td>
<td>50.49</td>
<td>9.72</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 4.7: Vector Quantization Compression Rate \( (\hat{r}_t) \) vs Granule

<table>
<thead>
<tr>
<th>Granule</th>
<th>( \hat{r}_t ) (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1.11</td>
<td>57.10</td>
<td>3.94</td>
<td>0.97</td>
</tr>
<tr>
<td>82</td>
<td>1.13</td>
<td>57.85</td>
<td>4.16</td>
<td>0.96</td>
</tr>
<tr>
<td>126</td>
<td>0.94</td>
<td>52.52</td>
<td>7.82</td>
<td>0.92</td>
</tr>
<tr>
<td>151</td>
<td>0.97</td>
<td>52.48</td>
<td>8.08</td>
<td>0.91</td>
</tr>
<tr>
<td>193</td>
<td>0.93</td>
<td>52.94</td>
<td>7.95</td>
<td>0.93</td>
</tr>
</tbody>
</table>

Tables 4.6 and 4.7 show the experimental compression rate \( (\hat{r}_t) \) obtained when using the previously specified parameters for each of the granules under study in case the prediction error is encoded using CELP and vector quantization respectively. Distortion is not only expressed as PSNR but also as the MPE corresponding to the reconstructed derived Level 2 products (surface pressure, surface temperature and total precipitable water) as well as the MSSIM coefficient obtained by averaging the individual coefficients that results from segmenting the image into \( 5 \times 5 \) windows as indicated in Section 1.3.2. Each pair of values
Figure 4.34: CELP: Original vs Reconstructed (PSNR = 50.49 dB), Granule 193

Figure 4.35: VQ: Original vs Reconstructed (PSNR = 52.94 dB), Granule 193
in the table is presented as a function of band ordering (BO) and subband decomposition (SD), the codebook size and the quantization resolution.

For both cases the experimental rates obtained for granules 126, 151 and 193 are within 10% of the calculated value. Granules 16 and 82 exhibit higher compression rate (within 25% of the calculated value) but considerable less distortion only in the case where vector quantization is used. For similar conditions CELP based encoding provides better compression rates but slightly more distortion that vector quantization based encoding of the prediction error.

Figures 4.34 and 4.35 shows bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 contrasting the original image (left) against the reconstructed one (right) for PSNR = 50.49 dB and PSNR = 52.94 dB corresponding to granule 193 for CELP and vector quantization encoding respectively. Pixels with a PMAD distortion larger than 10% are shown in color red.
Chapter 5

Rate Distortion Analysis

The previous chapters presented a new architecture for compression of ultraspectral data, specifically AIRS images, by means of novel preprocessing and compression stages. This chapter introduces a mathematical model to approximate the rate-distortion and compare it against the real performance of this new architecture.

5.1 Introduction

In order to model the rate-distortion the simplified block diagram shown in Figure 5.1 is used as reference. The AIRS input signal is initially subjected to band ordering (described in Section 3.3.2), subband decomposition (described in Section 3.3.1.2) and image data cube scanning (described in Section 3.3.3) as part of the preprocessing stage. The resulting speech-like signal is filtered by means of linear-prediction (described in Section 2.1.5) and the output error signal is encoded using a vector quantization scheme (described in Section 4.5). Similarly the set of linear prediction coefficients are encoded using a scalar quantization scheme (described in Section 4.4).

Each block in the diagram has parameters used to model its own behavior. Both the band ordering and the subband decomposition blocks are either enabled (on) or disabled (off). The scanning block is adjusted by defining the scanning block size ($WV$). The linear prediction filter block depends on the filter order $L$. The error encoding block involves a codebook where each codeword has $\nu$ components. On the other hand, the LP coefficient encoding block is ruled by a $q_\nu$-bit quantization resolution.

The rate-distortion model is based on the following assumptions:
The encoding of the prediction error is the only source of distortion. Assuming an open-loop error encoding scheme, distortion caused by LP coefficients quantization is masked by the prediction error distortion.

The effect of the band ordering and the subband decomposition is accounted by different values of the parameter of the Laplacian distribution of the error signal.

The error signal is partitioned into frames of size $v$ (as described in Section 4.5) before it is encoded.

## 5.2 Mathematical Model

This section deals with the mathematical models used to estimate the prediction error and represent the rate and the distortion as close expressions that can be compared against experimental data.

### 5.2.1 Prediction Error

As seen in Figure 4.30 the prediction error samples follow a Laplacian distribution with a probability density function that can be expressed as $f^j_x(x) = \frac{\lambda}{2} e^{-\lambda|x|}$. In general, for a given $j^{th}$ component $X_j$ of a codeword $X$, assuming a non-symmetric uniform quantizer of level $Q$ with reconstruction value at 0 that allows rates below one bit per sample it is possible to express the discrete probability distribution as a function of the interval $i$ by solving equation (5.1) below [67].

$$
P(X_j = i) = \int_{Q(i - \frac{1}{2})}^{Q(i + \frac{1}{2})} f^j_x(x) \, dx \tag{5.1}$$
Let’s define the following expressions

\[ \kappa_j = \lambda_j Q \]  

(5.2)

and

\[ \beta_j = \sinh \left( \frac{\kappa_j}{2} \right) \]  

(5.3)

where \( \lambda_j \) is the Laplacian parameter for the \( j^{th} \) component. Figure 5.2 shows the ratio \( \frac{\beta_j}{\kappa_j} \) as a function of \( \kappa_j \) to better understand their relationship which is bounded by \( 0 \leq \frac{\beta_j}{\kappa_j} \leq 2 \).

![Figure 5.2: \( \frac{\beta_j}{\kappa_j} \) vs \( \kappa_j \)]

Replacing expressions (5.2) and (5.3) in equation (5.1) discrete \( P(X_j = i) \) results as shown by equation (A.1) (see appendix A.1 for details).

\[
P(X_j = i) = \begin{cases} 
e^{-\kappa_j|i|} \beta_j & i \neq 0 \\ 1 - e^{-\frac{\kappa_j}{2}} & i = 0 \end{cases} \]  

(5.4)

Note that the exponential nature of the Laplacian distribution is preserved and for \( Q \gg 0 \) (high compression and high distortion) \( \kappa_j \gg 0 \) such that \( P(X_j = 0) \to 1 \) (all sample values are mapped to the same interval). On the other hand when \( Q \to 0 \) (low compression and
low distortion) $\kappa_j \to 0$ it is possible to expand $P(X_j = i)$ via Taylor series expansion and express it as $P(X_j = i) \approx \frac{\kappa_j}{2}$ (all sample values are uniformly distributed).

Although the prediction error tends to be approximately a white noise signal there is still a small correlation between samples that can be appreciated when examining the distribution of the current sample value given the previous one, namely $P(X_j|X_{j-1})$ assuming only a one sample dependency. Figure 5.3 shows the conditional distribution of the prediction error of the experimental data which follows a Van Houtum distribution which is a type of discrete quasi-uniform distribution defined by Equation (5.5) [64]. In this case the distribution parameters are $p_a = 0.09$, $p_b = 0.06$, $a = -1$ and $b = 1$.

$$P(X_j = i|X_{j-1} = k) = \begin{cases} 
    p_a & i = a + k \\
    p_b & i = b + k \\
    \frac{1-p_a-p_b}{b-a-1} & a + k < i < b + k \\
    0 & \text{otherwise}
\end{cases}$$

\text{(5.5)}

### 5.2.2 Rate

The rate (measured in bppb) induced by the transmission of a codeword (of size $v$) $X = (X_1, X_2 \cdots X_v)$, defined as $R_c$, is computed based on the ideal entropy of the source using
the chain rule and given by equation (5.6) [42].

$$R_c = \frac{1}{\upsilon} H(X) = \frac{1}{\upsilon} \sum_{j=1}^{\upsilon} H(X_j | X_{j-1} \cdots X_1)$$  \hspace{1cm} (5.6)$$

Note that typically the error prediction signal tends to be approximately a white noise signal where the error samples are independent of each other, however it can be assumed that there is still small correlation between samples as seen in Section 4.5 where it was shown that there is a one sample dependency such that $P(X_j, X_{j-1}) = P(X_j | X_{j-1}) P(X_{j-1})$. With these assumptions in place $H(X_j | X_{j-1} \cdots X_1) = H(X_j | X_{j-1})$ and the codeword rate can be considered as the aggregate of all individual sample rates $R_j$ as given by the expression (5.7) below.

$$R_c = \frac{1}{\upsilon} \sum_{j=1}^{\upsilon} H(X_j | X_{j-1}) = \frac{1}{\upsilon} \sum_{j=1}^{\upsilon} R_j$$ \hspace{1cm} (5.7)$$

Expanding $R_c$ it is possible to see that the rate $R_j$ is given by the expectation of the conditional probability as shown by equation (5.8).

$$R_j = -E[\log_2 P(X_j | X_{j-1})]$$ \hspace{1cm} (5.8)$$

For $j = 1$, $P(X_j | X_{j-1}) = P(X_j)$ such that when solving equation (5.8) based on the discrete distribution of the prediction error presented in expression (A.1), equation (5.9), which defines the $R_j$ as a function of $\kappa_j$ and $\beta_j$, is obtained (see appendix A.2 for details).

$$R_j = -\frac{1}{\log(2)} \left[ \log \left(1 - e^{-\frac{\kappa_j}{2}}\right) + e^{-\frac{\kappa_j}{2}} \log \left(\frac{1 + e^{-\frac{\kappa_j}{2}}}{2}\right) - \frac{\kappa_j}{2\beta_j} \right]$$ \hspace{1cm} (5.9)$$

Figure 5.4 shows $R_j$ as a function of $\kappa_j$ in semi logarithmic scale to remark the exponential dependency between the rate and the value of $\kappa_j$.

As $\kappa_j \to 0$ and $\beta_j \to 0$, $R_j \to \infty$ since there is an infinite number of quantization levels and therefore the transmission rate becomes infinite. In this case $R_c$ is ruled by the $\log \left(1 - e^{-\frac{\kappa_j}{2}}\right)$ in equation (5.9).

Similarly as $\kappa_j \to \infty$ and $\beta_j \to \infty$, $R_j \to 0$ because there is only one quantization level and no effective transmission is possible. In this case all terms in equation (5.9) are zero and in particular the last term because

$$\lim_{\kappa_j \to \infty} \frac{\kappa_j}{2\beta_j} = 0$$
where $\beta_j$ is a function of $\kappa_j$ as described in the previous subsection.

For $j > 1$, based on the discrete conditional distribution of the prediction error presented in expression (5.5), equation (5.10), which defines the $R_j$ as a function of $p_a$ and $p_b$, is obtained.

$$R_j = - [p_a \log_2 (p_a) + p_b \log_2 (p_b) + (1 - p_a - p_b) \log_2 (1 - p_a - p_b)] \quad (5.10)$$

Figure 5.5, again shown in a semilogarithmic scale, shows the overall transmission rate as a function of $\kappa_1$ when $v = 32$. This value is based on experimental data presented in Section 4.5 where the optimal partition frame size is obtained for a given prediction error signal. The reason $R_c$ is a function of $\kappa_j$ (with $j = 1$) is because in equation (5.7) $H(X_1)$ is the only member that is a function of $\kappa_j$ while all other terms $H(X_j|X_{j-1})$ (for $j > 1$) are independent of $\kappa_j$.

### 5.2.3 Distortion

If a codeword (of size $v$) $X = (X_1, X_2 \cdots X_v)$ is transmitted, the distortion for its $j^{th}$ component, based on the mean square error and defined as $D_j$, is given by the equation (5.11) below.
When replacing expression (A.1) into equation (5.11) equation (A.15), which defines the \( D_j \) as a function of the \( \kappa_j \) and \( \beta_j \), is obtained (see appendix A.3 for details).

\[
D_j = \sum_{i=-\infty}^{i=\infty} \int Q\left(i + \frac{1}{2}\right) Q\left(i - \frac{1}{2}\right) (x - Qi)^2 f_j(x) \, dx
\]  

(5.11)

The total distortion per pixel per band for any given codeword of size \( v \), defined as \( D_c \), is given by the expression (5.13).

\[
D_c = \frac{1}{v} \sum_{j=1}^{v} D_j
\]  

(5.13)

Equation (5.14) below presents the PSNR as a function of the distortion \( D_c \) where \( \text{MAX}_I \) is the maximum possible pixel value.

\[
\text{PSNR} = 10 \log_{10} \left( \frac{\text{MAX}_I^2}{D_c} \right)
\]  

(5.14)
Note that because
\[ \lim_{\kappa_j \to 0} \frac{\kappa_j}{\beta_j} = 2 \]
when \( \kappa_j \to 0 \) and \( \beta_j \to 0 \), \( D_j \to 0 \) since there is an infinite number of quantization levels and therefore the distortion becomes zero and the PSNR becomes infinite. Similarly as \( \kappa_j \to \infty \) and \( \beta_j \to \infty \), \( D_j \to 2 \lambda_j^2 \) because there is only one quantization level and the mean square rate distortion is a maximum equal to the corresponding Laplacian variance.

Without loss of generality it is possible to assume \( D_j \) independent of \( j \) such that each codeword sample follows the same independent Laplacian distribution, and the overall distortion is constant and equal to the sample distortion, namely \( D_c = D_j \). This fact can be seen in Figure 5.6 where \( D_c \) is shown as a function of \( \kappa_1 \) in a semi logarithmic scale (for \( \lambda = 40 \) based on the experimental data as indicated in Section 4.5). Both PSNR (shown in semilogarithmic scale in Figure 5.7) and \( R_c \) (shown in Figure 5.4) have a similar dependency with respect to \( \kappa_1 \), specifically PSNR decreases at a rate of 20 dB per decade and \( R_c \) decreases at a rate of 0.1 bppb per decade.

Figure 5.6: \( D_c \) vs \( \kappa_1 \) (\( \lambda = 40 \))
5.3 Theoretical Rate Distortion

The effect of band ordering and subband decomposition is taken into account by changing the Laplacian parameter $\lambda$, as indicated in Section 5.1. As shown in Figure 4.30 when band ordering is enabled and subband decomposition is disabled the value of the Laplacian parameter is typically $\lambda = 40$. Similarly, and based on experimental data, when both band ordering and subband decomposition are disabled the Laplacian parameter is $\lambda = 20$ and consequently the variance of the error is larger as shown in Figure 5.8. Additionally when subband decomposition is enabled the error distribution is further reduced as shown in Figures 5.9 and 5.10 for the cases where band ordering is disabled ($\lambda = 60$) and enabled respectively ($\lambda = 80$). Table 5.1 summarizes the value of the Laplacian parameter as a function of band ordering and subband decomposition.

<table>
<thead>
<tr>
<th>Band Ordering</th>
<th>Subband Decomposition</th>
<th>disabled</th>
<th>enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td></td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>enabled</td>
<td></td>
<td>60</td>
<td>80</td>
</tr>
</tbody>
</table>
Figure 5.8: Error Distribution: Actual Data vs Laplacian ($\lambda = 20$) (No Band Ordering / No Subband Decomposition)

Figure 5.9: Error Distribution: Actual Data vs Laplacian ($\lambda = 60$) (No Band Ordering / Subband Decomposition)
Figure 5.11 shows the rate-distortion curves obtained by plotting equation (5.14) against equation (5.7) when band ordering (BO) and subband decomposition (SD) are enabled or disabled. As indicated before the effect of these techniques are taken into account by modifying the Laplacian parameter value according to Table 5.1.

These four different possible combinations of band ordering and subband decomposition can be imposed to the experimental data in order to provide results than can be compared against the theoretical curves shown in Figure 5.11. Moreover the theoretical rate-distortion is obtained by varying the parameter $\kappa_1$, described in Section 5.2.1, in order to generate pair values of $R_c$, defined in equation (5.7) and $D_c$, defined in equation (5.13). Since $\kappa_1$ is proportional to $Q$, which is the quantization level, varying $\kappa_1$ implies varying $Q$ which indirectly implies varying $U$ defined as the codebook size in Section 4.5. Specifically by evaluating $U = 256, 512, 1024, 2048$ experimental values of rate-distortion are obtained.

Figures 5.12 through 5.15 plot the theoretical versus the experimental rate-distortion (for granule 193) for the four combinations of possible values of band ordering and subband decomposition.
5.4 Simplified Model

Note that for the most general case the idea is to provide improved compression and therefore minimize the codebook size while increasing the codeword length. This translates to \( Q_j \gg 0 \) which implies \( \kappa_j \gg 0 \) and \( \beta_j \gg 0 \) and manifests by approximating equation (5.9) by expression (5.15).

\[
R_j \approx \frac{\kappa_j}{2\beta_j \log(2)} \tag{5.15}
\]

When replacing equation (5.15) into equation (A.15), and assuming that the codeword components are independent of each other, a close form of the rate distortion curve is obtained. The component independence means that \( H (X_j|X_{j-1} \cdots X_1) = H (X_j) \) and therefore it is possible to express \( D_c \) as a function of \( R_c \) as shown in equation (5.16) below.

\[
D_c = \frac{2}{\lambda_j^2} [1 - R_c \log(2)] \tag{5.16}
\]

When combining expression (5.16) with the fact that \( D_c \gg 0 \) a bound for the rate \( R_c \) in equation (5.17) is found. This fact is based on the previous assumption that \( Q_j \gg 0 \) where high quantization levels imply very low rates and high distortion. Note that under these conditions the distortion is linearly dependent on the rate.
And a close form of PSNR vs \( R_c \) is obtained by putting expression (5.16) into equation (5.14) and generating the simplified rate distortion relationship shown in expression (5.18).

\[
\text{PSNR} = 10 \log_{10} \left( \frac{\text{MAX}_I^2}{\frac{2}{\lambda_i^2}(1 - R_c \log(2))} \right) 
\]

(5.18)

The variation of PSNR with respect to \( R_c \) is defined by the partial derivative \( \frac{\partial \text{PSNR}}{\partial R_c} \) which is given by the equation (5.19) below.
Figure 5.14: Theoretical vs Experimental Rate Distortion (No Band Ordering, Subband Decomposition)

Figure 5.15: Theoretical vs Experimental Rate Distortion (Band Ordering, Subband Decomposition)

$$\frac{\partial \text{PSNR}}{\partial R_c} = \frac{10 \log (2)}{\log (10) (1 - R_c \log (2))}$$  \hspace{1cm} (5.19)

When $R_c \rightarrow 0$, $\frac{\partial \text{PSNR}}{\partial R_c} \approx 3.0103$, which means that the variation of PSNR with respect to $R_c$ is around 3 dB per bppb indicating that although the initial variation is very small it tends to increase as the rate increases in part due to the term inside the denominator in the logarithm of equation (5.18) as seen in Figure 5.16. Similarly if the PSNR variation is about 100 dB for rates around the standard 1 bppb mark small rate variations can significantly improve or degrade the image quality.
5.5 Summary

This chapter presented a mathematical rate-distortion model of the preprocessing and compression stages that assumes that all distortion is ultimately exhibited by the error quantization and therefore can be estimated by calculating the mean square error as a function of the rate. Following the block diagram in Section 5.1 and adjusting results based on band ordering, subband decomposition and codeword size a series of parametric rate-distortion curves were obtained. The fundamental difference between curves arises from the using different values of the Laplacian parameter ($\lambda$) to represent the effects of band ordering and subband decomposition. When comparing the experimental data points (for granule 193) against the theoretical plots the results are pretty consistent and the calculated PSNR distortion is within 20% of the experimental value for a fixed rate value measured in bppb.
Chapter 6

Performance Comparison

This document has presented a new architecture for the compression of ultraspectral images but no comparison against the performance of other state of the art techniques has been discussed. This chapter deals with the performance improvements due to this new method, Niall introduced in [63], and compares the architecture against other known compression techniques, namely 3D ADPCM, JPEG-7, CALIC and pure subband decomposition as well as improvements to the latter like 3D SPIHT and 3D SPECK.

6.1 Introduction

The input data, to be used for the comparison between the different compression schemes, comes from granule 193 collected by NASA on March 2, 2004. It is important to mention that this image has been used as performance measure of other compression studies [16].

In Section 3.1 Figures 3.1 through 3.5 show eight representative bands of the original AIRS images corresponding to granules 16, 82, 126, 151 and 193 respectively. The images themselves will be subjected to the new compression architecture as well as compression by six other techniques already described in chapter 2. Specifically 3D ADPCM (described in section 2.3.5.1), JPEG-7 (described in section 2.3.5.2), CALIC (described in section 2.3.5.3), DWT, (described in section 2.1.3.3), 3D SPIHT and 3D SPECK (both also introduced in Section 2.1.3.3) will be compared against the AIRS LP compression technique presented earlier in this dissertation.

When the new AIRS LP compression is applied, the effect of different parameters including band ordering, subband decomposition, scanning block size, codebook size and filter
coefficients quantization will be evaluated. For all other compression techniques the distortion will be introduced through the quantization of the prediction error or the transform coding coefficients.

### 6.2 New AIRS LP CELP Compression

Figure 6.1 shows the block diagram for the new AIRS LP CELP-based compression prototype where the different parameters are adjusted to obtain empirical rate-distortion curves.

Table 6.1 enumerates the parameters that are evaluated as well as their range. Band ordering and Subband decomposition are either enabled or disabled. The scanning block size is $5 \times 5$ as it has been seen to provide the best experimental compression while preserving quality. The codebook size for the synthesis of the error signal is between 256 and 16384 codewords. The quantization levels for the filter coefficient encoding are either 32 or 128 per sample on a logarithmic scale (it is assumed that the coefficients follow a Laplacian distribution). Note that the linear coefficients are encoded only when the PMAD coefficient (as described in section 1.3.2) between successive error frames is greater than 10%.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Band Ordering</td>
<td>disabled, enabled</td>
</tr>
<tr>
<td>Subband Decomposition</td>
<td>disabled, enabled</td>
</tr>
<tr>
<td>Codebook Size</td>
<td>256, 1024, 4096, 16384</td>
</tr>
<tr>
<td>Quantizer [bits]</td>
<td>5, 7</td>
</tr>
</tbody>
</table>

Table 6.1: AIRS LP CELP Parameters
Table 6.2: AIRS LP CELP Data (Granule 16)

<table>
<thead>
<tr>
<th>BO and SD</th>
<th>Codebook</th>
<th>Q</th>
<th>rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>256</td>
<td>5</td>
<td>0.44</td>
<td>25.27</td>
<td>44.39</td>
<td>0.78</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>5</td>
<td>0.44</td>
<td>29.23</td>
<td>37.35</td>
<td>0.82</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>5</td>
<td>0.50</td>
<td>37.99</td>
<td>28.10</td>
<td>0.87</td>
</tr>
<tr>
<td>disabled</td>
<td>256</td>
<td>7</td>
<td>0.47</td>
<td>27.84</td>
<td>41.24</td>
<td>0.81</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>7</td>
<td>0.49</td>
<td>32.92</td>
<td>32.89</td>
<td>0.84</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>7</td>
<td>0.54</td>
<td>40.33</td>
<td>21.45</td>
<td>0.89</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>7</td>
<td>0.60</td>
<td>49.36</td>
<td>9.94</td>
<td>0.91</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>5</td>
<td>0.45</td>
<td>27.45</td>
<td>42.78</td>
<td>0.79</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>5</td>
<td>0.48</td>
<td>33.14</td>
<td>32.16</td>
<td>0.83</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>7</td>
<td>0.49</td>
<td>33.14</td>
<td>32.16</td>
<td>0.83</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>7</td>
<td>0.52</td>
<td>40.47</td>
<td>19.77</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 6.3: AIRS LP CELP Data (Granule 82)

<table>
<thead>
<tr>
<th>BO and SD</th>
<th>Codebook</th>
<th>Q</th>
<th>rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>256</td>
<td>5</td>
<td>0.44</td>
<td>23.20</td>
<td>47.20</td>
<td>0.58</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>5</td>
<td>0.55</td>
<td>30.52</td>
<td>35.43</td>
<td>0.78</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>5</td>
<td>0.50</td>
<td>39.11</td>
<td>22.71</td>
<td>0.82</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>5</td>
<td>0.56</td>
<td>45.76</td>
<td>10.34</td>
<td>0.88</td>
</tr>
<tr>
<td>disabled</td>
<td>256</td>
<td>7</td>
<td>0.52</td>
<td>33.96</td>
<td>39.17</td>
<td>0.70</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>7</td>
<td>0.58</td>
<td>39.27</td>
<td>33.30</td>
<td>0.79</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>7</td>
<td>0.62</td>
<td>48.41</td>
<td>11.31</td>
<td>0.90</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>5</td>
<td>0.46</td>
<td>26.36</td>
<td>44.50</td>
<td>0.61</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>5</td>
<td>0.50</td>
<td>34.91</td>
<td>30.29</td>
<td>0.76</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>5</td>
<td>0.55</td>
<td>44.13</td>
<td>19.35</td>
<td>0.80</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>5</td>
<td>0.66</td>
<td>51.26</td>
<td>8.05</td>
<td>0.93</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>7</td>
<td>0.54</td>
<td>34.54</td>
<td>29.48</td>
<td>0.78</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>7</td>
<td>0.58</td>
<td>42.03</td>
<td>20.75</td>
<td>0.84</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>7</td>
<td>0.60</td>
<td>49.39</td>
<td>10.62</td>
<td>0.89</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>7</td>
<td>0.70</td>
<td>61.66</td>
<td>1.37</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Table 6.4: AIRS LP CELP Data (Granule 126)

<table>
<thead>
<tr>
<th>BO and SD</th>
<th>Codebook</th>
<th>Q</th>
<th>rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>256</td>
<td>5</td>
<td>0.42</td>
<td>21.15</td>
<td>48.76</td>
<td>0.52</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>5</td>
<td>0.44</td>
<td>27.78</td>
<td>44.39</td>
<td>0.64</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>5</td>
<td>0.50</td>
<td>39.39</td>
<td>10.96</td>
<td>0.74</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>5</td>
<td>0.61</td>
<td>39.40</td>
<td>22.12</td>
<td>0.79</td>
</tr>
<tr>
<td>disabled</td>
<td>256</td>
<td>7</td>
<td>0.47</td>
<td>23.25</td>
<td>47.57</td>
<td>0.71</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>7</td>
<td>0.53</td>
<td>39.78</td>
<td>23.88</td>
<td>0.80</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>7</td>
<td>0.53</td>
<td>39.78</td>
<td>23.88</td>
<td>0.80</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>5</td>
<td>0.43</td>
<td>24.74</td>
<td>48.45</td>
<td>0.64</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>5</td>
<td>0.47</td>
<td>27.85</td>
<td>44.39</td>
<td>0.64</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>5</td>
<td>0.53</td>
<td>39.78</td>
<td>21.18</td>
<td>0.82</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>5</td>
<td>0.63</td>
<td>51.74</td>
<td>21.18</td>
<td>0.93</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>7</td>
<td>0.46</td>
<td>25.91</td>
<td>45.90</td>
<td>0.80</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>7</td>
<td>0.54</td>
<td>34.38</td>
<td>29.77</td>
<td>0.78</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>7</td>
<td>0.59</td>
<td>43.08</td>
<td>16.44</td>
<td>0.88</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>7</td>
<td>0.69</td>
<td>56.26</td>
<td>9.10</td>
<td>0.93</td>
</tr>
</tbody>
</table>
### Table 6.5: AIRS LP CELP Data (Granule 151)

<table>
<thead>
<tr>
<th>BO and SD</th>
<th>Codebook</th>
<th>Q</th>
<th>rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>256</td>
<td>5</td>
<td>0.42</td>
<td>21.66</td>
<td>49.92</td>
<td>0.51</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>5</td>
<td>0.44</td>
<td>27.71</td>
<td>42.61</td>
<td>0.67</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>5</td>
<td>0.52</td>
<td>32.84</td>
<td>38.32</td>
<td>0.72</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>5</td>
<td>0.64</td>
<td>41.48</td>
<td>24.880.81</td>
<td></td>
</tr>
<tr>
<td>disabled</td>
<td>256</td>
<td>7</td>
<td>0.42</td>
<td>24.84</td>
<td>46.52</td>
<td>0.62</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>7</td>
<td>0.51</td>
<td>32.41</td>
<td>46.09</td>
<td>0.72</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>7</td>
<td>0.55</td>
<td>40.15</td>
<td>24.84</td>
<td>0.72</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>7</td>
<td>0.56</td>
<td>52.57</td>
<td>8.62</td>
<td>0.93</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>5</td>
<td>0.43</td>
<td>23.82</td>
<td>45.50</td>
<td>0.60</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>5</td>
<td>0.48</td>
<td>30.55</td>
<td>40.17</td>
<td>0.70</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>5</td>
<td>0.56</td>
<td>38.53</td>
<td>26.29</td>
<td>0.78</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>5</td>
<td>0.68</td>
<td>49.86</td>
<td>10.03</td>
<td>0.90</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>7</td>
<td>0.51</td>
<td>27.16</td>
<td>42.41</td>
<td>0.88</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>7</td>
<td>0.52</td>
<td>36.06</td>
<td>40.01</td>
<td>0.78</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>7</td>
<td>0.60</td>
<td>48.05</td>
<td>12.14</td>
<td>0.88</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>7</td>
<td>0.71</td>
<td>56.13</td>
<td>3.73</td>
<td>0.98</td>
</tr>
</tbody>
</table>

### Table 6.6: AIRS LP CELP Data (Granule 193)

<table>
<thead>
<tr>
<th>BO and SD</th>
<th>Codebook</th>
<th>Q</th>
<th>rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>256</td>
<td>5</td>
<td>0.42</td>
<td>21.85</td>
<td>50.19</td>
<td>0.49</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>5</td>
<td>0.45</td>
<td>26.26</td>
<td>45.74</td>
<td>0.84</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>5</td>
<td>0.54</td>
<td>31.67</td>
<td>39.10</td>
<td>0.74</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>5</td>
<td>0.56</td>
<td>40.79</td>
<td>24.77</td>
<td>0.80</td>
</tr>
<tr>
<td>disabled</td>
<td>256</td>
<td>7</td>
<td>0.43</td>
<td>23.25</td>
<td>48.06</td>
<td>0.54</td>
</tr>
<tr>
<td>disabled</td>
<td>1024</td>
<td>7</td>
<td>0.50</td>
<td>29.47</td>
<td>41.09</td>
<td>0.75</td>
</tr>
<tr>
<td>disabled</td>
<td>4096</td>
<td>7</td>
<td>0.56</td>
<td>38.04</td>
<td>26.45</td>
<td>0.79</td>
</tr>
<tr>
<td>disabled</td>
<td>16384</td>
<td>7</td>
<td>0.58</td>
<td>50.49</td>
<td>9.60</td>
<td>0.91</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>5</td>
<td>0.44</td>
<td>21.93</td>
<td>45.85</td>
<td>0.53</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>5</td>
<td>0.49</td>
<td>28.56</td>
<td>43.16</td>
<td>0.68</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>5</td>
<td>0.57</td>
<td>28.25</td>
<td>28.96</td>
<td>0.77</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>5</td>
<td>0.61</td>
<td>47.95</td>
<td>13.24</td>
<td>0.93</td>
</tr>
<tr>
<td>enabled</td>
<td>256</td>
<td>7</td>
<td>0.65</td>
<td>26.25</td>
<td>54.10</td>
<td>0.61</td>
</tr>
<tr>
<td>enabled</td>
<td>1024</td>
<td>7</td>
<td>0.65</td>
<td>34.37</td>
<td>31.48</td>
<td>0.74</td>
</tr>
<tr>
<td>enabled</td>
<td>4096</td>
<td>7</td>
<td>0.61</td>
<td>50.04</td>
<td>10.73</td>
<td>0.90</td>
</tr>
<tr>
<td>enabled</td>
<td>16384</td>
<td>7</td>
<td>0.65</td>
<td>56.31</td>
<td>4.68</td>
<td>0.98</td>
</tr>
</tbody>
</table>

143
Figure 6.2: Original vs Reconstructed (PSNR = 33.14 dB), Granule 16

Figure 6.3: Original vs Reconstructed (PSNR = 40.77 dB), Granule 16
Figure 6.4: Original vs Reconstructed (PSNR = 47.98 dB), Granule 16

Figure 6.5: Original vs Reconstructed (PSNR = 60.55 dB), Granule 16
Figure 6.6: Original vs Reconstructed (PSNR = 34.54 dB), Granule 82

Figure 6.7: Original vs Reconstructed (PSNR = 42.03 dB), Granule 82
Figure 6.8: Original vs Reconstructed (PSNR = 49.39 dB), Granule 82

Figure 6.9: Original vs Reconstructed (PSNR = 61.66 dB), Granule 82
Figure 6.10: Original vs Reconstructed (PSNR = 25.91 dB), Granule 126

Figure 6.11: Original vs Reconstructed (PSNR = 34.38 dB), Granule 126
Figure 6.12: Original vs Reconstructed (PSNR = 43.08 dB), Granule 126

Figure 6.13: Original vs Reconstructed (PSNR = 55.26 dB), Granule 126
Figure 6.14: Original vs Reconstructed (PSNR = 27.16 dB), Granule 151

Figure 6.15: Original vs Reconstructed (PSNR = 36.05 dB), Granule 151
Figure 6.16: Original vs Reconstructed (PSNR = 44.05 dB), Granule 151

Figure 6.17: Original vs Reconstructed (PSNR = 56.13 dB), Granule 151
Tables 6.2 through 6.6 show the rate and the distortion for granules 16 through 193 respectively. In addition to PSNR, the MPE corresponding to the reconstructed derived Level 2 products (surface pressure, surface temperature and total precipitable water) as well as the MSSIM coefficient obtained by averaging the individual coefficients that results from segmenting the image into $5 \times 5$ windows as indicated in Section 1.3.2. Each pair of values in the table is presented as a function of band ordering (BO) and subband decomposition (SD), the codebook size and the quantization resolution. Similarly Figures 6.2 through 6.21 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for different distortion levels. Pixels with a PMAD distortion larger than 10% are shown in color red. As expected visual inspection shows minimal distortion for PSNR = 61.66 dB. Figures 6.22 through 6.26 plot the rate-distortion relationship for granules 16 through 193 respectively.

### 6.3 Other Compression Techniques

This section describes the results obtained when applying other well-known compression techniques to the reference input data.
Figure 6.19: Original vs Reconstructed (PSNR = 34.47 dB), Granule 193

Figure 6.20: Original vs Reconstructed (PSNR = 43.04 dB), Granule 193
Figure 6.21: Original vs Reconstructed (PSNR = 55.31 dB), Granule 193

Figure 6.22: Experimental Rate-Distortion (Granule 16)
Figure 6.23: Experimental Rate-Distortion (Granule 82)

Figure 6.24: Experimental Rate-Distortion (Granule 126)
Figure 6.25: Experimental Rate-Distortion (Granule 151)

Figure 6.26: Experimental Rate-Distortion (Granule 193)
6.3.1 3D ADPCM

Under 3D ADPCM, described in Section 2.3.5.1, different fixed linear prediction schemes are used and the prediction error is encoded using Rice coding. Scalar linear quantization of the error signal provides different combinations of rate-distortion values as shown in Tables 6.7, 6.8 and 6.9 for a 6-bit, 7-bit and 8-bit quantization resolution respectively.

Table 6.7: 3D APDCM (Q = 6)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1.56</td>
<td>36.81</td>
<td>29.22</td>
<td>0.76</td>
</tr>
<tr>
<td>82</td>
<td>1.75</td>
<td>35.59</td>
<td>30.76</td>
<td>0.75</td>
</tr>
<tr>
<td>126</td>
<td>1.89</td>
<td>40.15</td>
<td>23.81</td>
<td>0.80</td>
</tr>
<tr>
<td>151</td>
<td>1.86</td>
<td>39.91</td>
<td>23.55</td>
<td>0.79</td>
</tr>
<tr>
<td>193</td>
<td>1.91</td>
<td>40.11</td>
<td>22.04</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Table 6.8: 3D APDCM (Q = 7)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>2.56</td>
<td>44.54</td>
<td>16.71</td>
<td>0.85</td>
</tr>
<tr>
<td>82</td>
<td>2.75</td>
<td>43.10</td>
<td>17.28</td>
<td>0.84</td>
</tr>
<tr>
<td>126</td>
<td>2.89</td>
<td>47.22</td>
<td>13.84</td>
<td>0.87</td>
</tr>
<tr>
<td>151</td>
<td>2.86</td>
<td>47.48</td>
<td>12.49</td>
<td>0.88</td>
</tr>
<tr>
<td>193</td>
<td>2.91</td>
<td>47.57</td>
<td>12.56</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 6.9: 3D APDCM (Q = 8)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>3.56</td>
<td>80.61</td>
<td>0.10</td>
<td>0.99</td>
</tr>
<tr>
<td>82</td>
<td>3.75</td>
<td>76.87</td>
<td>0.45</td>
<td>0.98</td>
</tr>
<tr>
<td>126</td>
<td>3.89</td>
<td>79.24</td>
<td>0.26</td>
<td>0.99</td>
</tr>
<tr>
<td>151</td>
<td>3.86</td>
<td>77.85</td>
<td>0.33</td>
<td>0.99</td>
</tr>
<tr>
<td>193</td>
<td>3.91</td>
<td>78.08</td>
<td>0.31</td>
<td>0.99</td>
</tr>
</tbody>
</table>
Figure 6.27: Original vs Reconstructed (PSNR = 36.81 dB), Granule 16 (Q = 6)

Figure 6.28: Original vs Reconstructed (PSNR = 35.59 dB), Granule 82 (Q = 6)
Figure 6.29: Original vs Reconstructed (PSNR = 40.15 dB), Granule 126 (Q = 6)

Figure 6.30: Original vs Reconstructed (PSNR = 39.91 dB), Granule 151 (Q = 6)
Figure 6.31: Original vs Reconstructed (PSNR = 40.11 dB), Granule 193 (Q = 6)

Figure 6.32: Original vs Reconstructed (PSNR = 44.54 dB), Granule 16 (Q = 7)
Figure 6.33: Original vs Reconstructed (PSNR = 43.10 dB), Granule 82 (Q = 7)

Figure 6.34: Original vs Reconstructed (PSNR = 47.22 dB), Granule 126 (Q = 7)
Figure 6.35: Original vs Reconstructed (PSNR = 47.48 dB), Granule 151 (Q = 7)

Figure 6.36: Original vs Reconstructed (PSNR = 47.57 dB), Granule 193 (Q = 7)
Figures 6.27 through 6.36 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for each of the combinations in Tables 6.7 through 6.9. Each reconstructed image is compared against the original image and as before pixels with a PMAD distortion larger than 10% shown in color red.

### 6.3.2 JPEG-7

*JPEG-7*, described in Section 2.3.5.2, applies different fixed linear prediction schemes (different than the ones for 3D ADPCM) and encodes the prediction error using regular Huffman entropy coding. As in the previous case different scalar linear quantization of the error signal provides different combinations of rate-distortion values as shown in Tables 6.10 through 6.14 for quantization level 4-bit through 8-bit respectively.

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1.34</td>
<td>35.70</td>
<td>30.11</td>
<td>0.75</td>
</tr>
<tr>
<td>82</td>
<td>1.78</td>
<td>36.67</td>
<td>29.48</td>
<td>0.76</td>
</tr>
<tr>
<td>126</td>
<td>1.91</td>
<td>34.56</td>
<td>31.77</td>
<td>0.74</td>
</tr>
<tr>
<td>151</td>
<td>1.82</td>
<td>37.76</td>
<td>27.52</td>
<td>0.77</td>
</tr>
<tr>
<td>193</td>
<td>1.95</td>
<td>34.86</td>
<td>31.20</td>
<td>0.74</td>
</tr>
</tbody>
</table>

Figures 6.42 through 6.66 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000
Figure 6.38: Original vs Reconstructed (PSNR = 76.87 dB), Granule 82 (Q = 8)

Figure 6.39: Original vs Reconstructed (PSNR = 79.24 dB), Granule 126 (Q = 8)
Figure 6.40: Original vs Reconstructed (PSNR = 77.85 dB), Granule 151 (Q = 8)

Figure 6.41: Original vs Reconstructed (PSNR = 78.08 dB), Granule 193 (Q = 8)
Table 6.11: JPEG-7 (Q = 5)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>2.19</td>
<td>40.89</td>
<td>23.92</td>
<td>0.79</td>
</tr>
<tr>
<td>82</td>
<td>2.54</td>
<td>42.51</td>
<td>19.65</td>
<td>0.81</td>
</tr>
<tr>
<td>126</td>
<td>2.83</td>
<td>40.26</td>
<td>22.52</td>
<td>0.80</td>
</tr>
<tr>
<td>151</td>
<td>2.62</td>
<td>43.59</td>
<td>19.23</td>
<td>0.81</td>
</tr>
<tr>
<td>193</td>
<td>2.78</td>
<td>40.69</td>
<td>21.09</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Table 6.12: JPEG-7 (Q = 6)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>3.33</td>
<td>47.59</td>
<td>12.79</td>
<td>0.87</td>
</tr>
<tr>
<td>82</td>
<td>3.68</td>
<td>49.25</td>
<td>10.04</td>
<td>0.90</td>
</tr>
<tr>
<td>126</td>
<td>3.96</td>
<td>46.97</td>
<td>14.39</td>
<td>0.86</td>
</tr>
<tr>
<td>151</td>
<td>3.77</td>
<td>49.99</td>
<td>11.62</td>
<td>0.90</td>
</tr>
<tr>
<td>193</td>
<td>3.91</td>
<td>47.27</td>
<td>12.31</td>
<td>0.88</td>
</tr>
</tbody>
</table>

comparing the original image against the reconstructed one for each of the combinations in Tables 6.10 through 6.14. Each reconstructed image is compared against the original image and as before pixels with a PMAD distortion larger than 10% shown in color red. Heavy distortion can be found for 4-bit and 5-bit quantization.

Table 6.13: JPEG-7 (Q = 7)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4.45</td>
<td>55.60</td>
<td>5.09</td>
<td>0.95</td>
</tr>
<tr>
<td>82</td>
<td>4.79</td>
<td>57.97</td>
<td>4.21</td>
<td>0.94</td>
</tr>
<tr>
<td>126</td>
<td>5.05</td>
<td>55.57</td>
<td>5.14</td>
<td>0.95</td>
</tr>
<tr>
<td>151</td>
<td>4.88</td>
<td>57.76</td>
<td>4.59</td>
<td>0.94</td>
</tr>
<tr>
<td>193</td>
<td>5.01</td>
<td>55.43</td>
<td>4.98</td>
<td>0.94</td>
</tr>
</tbody>
</table>
Figure 6.42: Original vs Reconstructed (PSNR = 35.70 dB), Granule 16 (Q = 4)

Figure 6.43: Original vs Reconstructed (PSNR = 36.67 dB), Granule 82 (Q = 4)
Figure 6.44: Original vs Reconstructed (PSNR = 34.56 dB), Granule 126 (Q = 4)

Figure 6.45: Original vs Reconstructed (PSNR = 37.76 dB), Granule 151 (Q = 4)
Figure 6.46: Original vs Reconstructed (PSNR = 34.86 dB), Granule 193 (Q = 4)

Figure 6.47: Original vs Reconstructed (PSNR = 40.89 dB), Granule 16 (Q = 5)
Figure 6.48: Original vs Reconstructed (PSNR = 42.51 dB), Granule 82 (Q = 5)

Figure 6.49: Original vs Reconstructed (PSNR = 40.26 dB), Granule 126 (Q = 5)
Figure 6.50: Original vs Reconstructed (PSNR = 43.59 dB), Granule 151 (Q = 5)

Figure 6.51: Original vs Reconstructed (PSNR = 40.69 dB), Granule 193 (Q = 5)
Figure 6.52: Original vs Reconstructed (PSNR = 47.59 dB), Granule 16 (Q = 6)

Figure 6.53: Original vs Reconstructed (PSNR = 49.25 dB), Granule 82 (Q = 6)
Figure 6.54: Original vs Reconstructed (PSNR = 46.97 dB), Granule 126 (Q = 6)

Figure 6.55: Original vs Reconstructed (PSNR = 49.99 dB), Granule 151 (Q = 6)
Table 6.14: JPEG-7 (Q = 8)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>5.50</td>
<td>121.84</td>
<td>0.02</td>
<td>0.99</td>
</tr>
<tr>
<td>82</td>
<td>5.83</td>
<td>94.05</td>
<td>0.05</td>
<td>0.99</td>
</tr>
<tr>
<td>126</td>
<td>6.09</td>
<td>99.61</td>
<td>0.03</td>
<td>0.99</td>
</tr>
<tr>
<td>151</td>
<td>5.92</td>
<td>87.04</td>
<td>0.08</td>
<td>0.99</td>
</tr>
<tr>
<td>193</td>
<td>6.05</td>
<td>87.28</td>
<td>0.09</td>
<td>0.99</td>
</tr>
</tbody>
</table>
6.3.3 CALIC

Under CALIC, described in Section 2.3.5.3, different fixed prediction schemes are applied while the prediction error is encoded using regular Huffman entropy coding based on a context derived from the gradient calculation for any given pixel. Different scalar linear quantization of the error signal provides different combinations of rate-distortion values as shown by Tables 6.15 through 6.19.

Table 6.15: CALIC (Q = 2)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0.95</td>
<td>33.25</td>
<td>35.14</td>
<td>0.73</td>
</tr>
<tr>
<td>82</td>
<td>1.47</td>
<td>33.24</td>
<td>34.95</td>
<td>0.72</td>
</tr>
<tr>
<td>126</td>
<td>1.45</td>
<td>33.20</td>
<td>35.01</td>
<td>0.72</td>
</tr>
<tr>
<td>151</td>
<td>1.35</td>
<td>33.19</td>
<td>35.59</td>
<td>0.72</td>
</tr>
<tr>
<td>193</td>
<td>1.31</td>
<td>33.14</td>
<td>35.35</td>
<td>0.72</td>
</tr>
</tbody>
</table>

Figures 6.67 through 6.91 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for each of the combinations in Tables 6.15 through 6.19. Each reconstructed image is compared against the original image and as before pixels with a PMAD distortion larger than 10% shown in color red. Heavy distortion can be found for 2-bit quantization.
Figure 6.57: Original vs Reconstructed (PSNR = 55.60 dB), Granule 16 (Q = 7)

Figure 6.58: Original vs Reconstructed (PSNR = 57.97 dB), Granule 82 (Q = 7)
Figure 6.59: Original vs Reconstructed (PSNR = 55.57 dB), Granule 126 (Q = 7)

Figure 6.60: Original vs Reconstructed (PSNR = 57.76 dB), Granule 151 (Q = 7)
Figure 6.61: Original vs Reconstructed (PSNR = 55.43 dB), Granule 193 (Q = 7)

Figure 6.62: Original vs Reconstructed (PSNR = 121.84 dB), Granule 16 (Q = 8)
Figure 6.63: Original vs Reconstructed (PSNR = 94.05 dB), Granule 82 (Q = 8)

Figure 6.64: Original vs Reconstructed (PSNR = 99.61 dB), Granule 126 (Q = 8)
Figure 6.65: Original vs Reconstructed (PSNR = 87.04 dB), Granule 151 (Q = 8)

Figure 6.66: Original vs Reconstructed (PSNR = 87.28 dB), Granule 193 (Q = 8)
Table 6.16: CALIC (Q = 3)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1.72</td>
<td>39.38</td>
<td>21.98</td>
<td>0.84</td>
</tr>
<tr>
<td>82</td>
<td>2.26</td>
<td>39.26</td>
<td>22.03</td>
<td>0.81</td>
</tr>
<tr>
<td>126</td>
<td>2.20</td>
<td>39.18</td>
<td>23.09</td>
<td>0.80</td>
</tr>
<tr>
<td>151</td>
<td>2.02</td>
<td>39.18</td>
<td>22.44</td>
<td>0.82</td>
</tr>
<tr>
<td>193</td>
<td>2.06</td>
<td>39.18</td>
<td>22.20</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Table 6.17: CALIC (Q = 4)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>2.67</td>
<td>44.60</td>
<td>16.21</td>
<td>0.86</td>
</tr>
<tr>
<td>82</td>
<td>3.21</td>
<td>44.21</td>
<td>16.43</td>
<td>0.86</td>
</tr>
<tr>
<td>126</td>
<td>3.15</td>
<td>44.10</td>
<td>17.93</td>
<td>0.85</td>
</tr>
<tr>
<td>151</td>
<td>2.97</td>
<td>44.19</td>
<td>16.26</td>
<td>0.86</td>
</tr>
<tr>
<td>193</td>
<td>3.00</td>
<td>44.14</td>
<td>17.51</td>
<td>0.87</td>
</tr>
</tbody>
</table>

Table 6.18: CALIC (Q = 5)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>3.65</td>
<td>47.22</td>
<td>11.90</td>
<td>0.91</td>
</tr>
<tr>
<td>82</td>
<td>4.18</td>
<td>46.66</td>
<td>12.05</td>
<td>0.92</td>
</tr>
<tr>
<td>126</td>
<td>4.13</td>
<td>46.53</td>
<td>12.53</td>
<td>0.90</td>
</tr>
<tr>
<td>151</td>
<td>3.94</td>
<td>46.68</td>
<td>13.30</td>
<td>0.89</td>
</tr>
<tr>
<td>193</td>
<td>3.97</td>
<td>46.60</td>
<td>12.11</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 6.19: CALIC (Q = 6)

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4.63</td>
<td>47.71</td>
<td>11.04</td>
<td>0.90</td>
</tr>
<tr>
<td>82</td>
<td>5.15</td>
<td>47.32</td>
<td>11.24</td>
<td>0.89</td>
</tr>
<tr>
<td>126</td>
<td>5.09</td>
<td>47.18</td>
<td>11.19</td>
<td>0.91</td>
</tr>
<tr>
<td>151</td>
<td>4.91</td>
<td>47.26</td>
<td>10.98</td>
<td>0.91</td>
</tr>
<tr>
<td>193</td>
<td>4.94</td>
<td>47.22</td>
<td>10.75</td>
<td>0.89</td>
</tr>
</tbody>
</table>
Figure 6.67: Original vs Reconstructed (PSNR = 33.25 dB), Granule 16 (Q = 2)

Figure 6.68: Original vs Reconstructed (PSNR = 33.24 dB), Granule 82 (Q = 2)
Figure 6.69: Original vs Reconstructed (PSNR = 33.20 dB), Granule 126 (Q = 2)

Figure 6.70: Original vs Reconstructed (PSNR = 33.19 dB), Granule 151 (Q = 2)
Figure 6.71: Original vs Reconstructed (PSNR = 33.14 dB), Granule 193 (Q = 2)

Figure 6.72: Original vs Reconstructed (PSNR = 39.38 dB), Granule 16 (Q = 3)
Figure 6.73: Original vs Reconstructed (PSNR = 39.26 dB), Granule 82 (Q = 3)

Figure 6.74: Original vs Reconstructed (PSNR = 39.18 dB), Granule 126 (Q = 3)
Figure 6.75: Original vs Reconstructed (PSNR = 39.18 dB), Granule 151 (Q = 3)

Figure 6.76: Original vs Reconstructed (PSNR = 39.18 dB), Granule 193 (Q = 3)
Figure 6.77: Original vs Reconstructed (PSNR = 44.60 dB), Granule 16 (Q = 4)

Figure 6.78: Original vs Reconstructed (PSNR = 44.21 dB), Granule 82 (Q = 4)
Figure 6.79: Original vs Reconstructed (PSNR = 44.10 dB), Granule 126 (Q = 4)

Figure 6.80: Original vs Reconstructed (PSNR = 44.19 dB), Granule 151 (Q = 4)
Figure 6.81: Original vs Reconstructed (PSNR = 44.14 dB), Granule 193 (Q = 4)

Figure 6.82: Original vs Reconstructed (PSNR = 47.22 dB), Granule 16 (Q = 5)
Figure 6.83: Original vs Reconstructed (PSNR = 46.66 dB), Granule 82 (Q = 5)

Figure 6.84: Original vs Reconstructed (PSNR = 46.53 dB), Granule 126 (Q = 5)
Figure 6.85: Original vs Reconstructed (PSNR = 46.68 dB), Granule 151 (Q = 5)

6.3.4 DWT

DWT, described in Section 2.1.3.3, uses subband decomposition to accomplish compression. When the different bands are quantized at different quantization levels, different combinations of rate-distortion values are obtained as shown in Tables 6.20 through 6.24 for granules 16, 82, 126, 151 and 193 respectively.

<table>
<thead>
<tr>
<th>Q22</th>
<th>Q21</th>
<th>Q1</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1.75</td>
<td>40.33</td>
<td>24.51</td>
<td>0.61</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
<td>44.90</td>
<td>16.10</td>
<td>0.82</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3.75</td>
<td>51.66</td>
<td>9.83</td>
<td>0.91</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4.75</td>
<td>54.55</td>
<td>5.59</td>
<td>0.96</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5.75</td>
<td>57.21</td>
<td>3.93</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Figures 6.92 through 6.116 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for each of the combinations in Tables 6.20 through 6.24. Heavy distortion can be found for low quantization levels.
Figure 6.86: Original vs Reconstructed (PSNR = 46.60 dB), Granule 193 (Q = 5)

Figure 6.87: Original vs Reconstructed (PSNR = 47.71 dB), Granule 16 (Q = 6)
Figure 6.88: Original vs Reconstructed (PSNR = 47.32 dB), Granule 82 (Q = 6)

Figure 6.89: Original vs Reconstructed (PSNR = 47.18 dB), Granule 126 (Q = 6)
Figure 6.90: Original vs Reconstructed (PSNR = 47.26 dB), Granule 151 (Q = 6)

Figure 6.91: Original vs Reconstructed (PSNR = 47.22 dB), Granule 193 (Q = 6)
Table 6.21: DWT (Granule 82)

<table>
<thead>
<tr>
<th>Q22</th>
<th>Q21</th>
<th>Q1</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1.75</td>
<td>30.96</td>
<td>40.19</td>
<td>0.70</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
<td>36.68</td>
<td>29.61</td>
<td>0.74</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3.75</td>
<td>43.55</td>
<td>18.82</td>
<td>0.82</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4.75</td>
<td>45.92</td>
<td>15.63</td>
<td>0.87</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5.75</td>
<td>49.33</td>
<td>10.05</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 6.22: DWT (Granule 126)

<table>
<thead>
<tr>
<th>Q22</th>
<th>Q21</th>
<th>Q1</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1.75</td>
<td>35.47</td>
<td>31.62</td>
<td>0.75</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
<td>39.97</td>
<td>22.09</td>
<td>0.81</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3.75</td>
<td>44.42</td>
<td>15.12</td>
<td>0.86</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4.75</td>
<td>47.60</td>
<td>11.74</td>
<td>0.89</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5.75</td>
<td>49.53</td>
<td>10.62</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 6.23: DWT (Granule 151)

<table>
<thead>
<tr>
<th>Q22</th>
<th>Q21</th>
<th>Q1</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1.75</td>
<td>35.53</td>
<td>30.01</td>
<td>0.75</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
<td>40.50</td>
<td>24.72</td>
<td>0.77</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3.75</td>
<td>43.05</td>
<td>19.50</td>
<td>0.83</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4.75</td>
<td>47.07</td>
<td>13.38</td>
<td>0.87</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5.75</td>
<td>48.95</td>
<td>12.82</td>
<td>0.88</td>
</tr>
</tbody>
</table>

Table 6.24: DWT (Granule 193)

<table>
<thead>
<tr>
<th>Q22</th>
<th>Q21</th>
<th>Q1</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1.75</td>
<td>34.54</td>
<td>31.38</td>
<td>0.71</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2.75</td>
<td>38.66</td>
<td>24.79</td>
<td>0.78</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3.75</td>
<td>42.79</td>
<td>20.12</td>
<td>0.84</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4.75</td>
<td>46.51</td>
<td>14.82</td>
<td>0.86</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5.75</td>
<td>48.66</td>
<td>12.09</td>
<td>0.88</td>
</tr>
</tbody>
</table>
Figure 6.92: Original vs Reconstructed (PSNR = 40.33 dB), Granule 16 (Q1 = 2, Q21 = 3, Q22 = 4)

Figure 6.93: Original vs Reconstructed (PSNR = 30.96 dB), Granule 82 (Q1 = 2, Q21 = 3, Q22 = 4)
Figure 6.94: Original vs Reconstructed (PSNR = 35.47 dB), Granule 126 (Q1 = 2, Q21 = 3, Q22 = 4)

Figure 6.95: Original vs Reconstructed (PSNR = 35.53 dB), Granule 151 (Q1 = 2, Q21 = 3, Q22 = 4)
Figure 6.96: Original vs Reconstructed (PSNR = 34.54 dB), Granule 193 (Q1 = 2, Q21 = 3, Q22 = 4)

Figure 6.97: Original vs Reconstructed (PSNR = 44.90 dB), Granule 16 (Q1 = 3, Q21 = 4, Q22 = 5)
Figure 6.98: Original vs Reconstructed (PSNR = 36.68 dB), Granule 82 (Q1 = 3, Q21 = 4, Q22 = 5)

Figure 6.99: Original vs Reconstructed (PSNR = 39.97 dB), Granule 126 (Q1 = 3, Q21 = 4, Q22 = 5)
Figure 6.100: Original vs Reconstructed (PSNR = 40.50 dB), Granule 151 (Q1 = 3, Q21 = 4, Q22 = 5)

Figure 6.101: Original vs Reconstructed (PSNR = 38.66 dB), Granule 193 (Q1 = 3, Q21 = 4, Q22 = 5)
Figure 6.102: Original vs Reconstructed (PSNR = 51.66 dB), Granule 16 (Q1 = 4, Q21 = 5, Q22 = 6)

Figure 6.103: Original vs Reconstructed (PSNR = 43.55 dB), Granule 82 (Q1 = 4, Q21 = 5, Q22 = 6)
Figure 6.104: Original vs Reconstructed (PSNR = 44.42 dB), Granule 126 (Q1 = 4, Q21 = 5, Q22 = 6)

Figure 6.105: Original vs Reconstructed (PSNR = 43.05 dB), Granule 151 (Q1 = 4, Q21 = 5, Q22 = 6)
Figure 6.106: Original vs Reconstructed (PSNR = 42.79 dB), Granule 193 (Q1 = 4, Q21 = 5, Q22 = 6)

Figure 6.107: Original vs Reconstructed (PSNR = 54.55 dB), Granule 16 (Q1 = 5, Q21 = 6, Q22 = 7)
Figure 6.108: Original vs Reconstructed (PSNR = 45.92 dB), Granule 82 (Q1 = 5, Q21 = 6, Q22 = 7)

Figure 6.109: Original vs Reconstructed (PSNR = 47.60 dB), Granule 126 (Q1 = 5, Q21 = 6, Q22 = 7)
6.3.5 3D SPIHT

3D SPIHT is the multispectral extension of SPIHT, mentioned in Section 2.1.3.3, which is essentially a wavelet transform-based embedded bit-plane encoding technique. SPIHT partitions transformed DWT coefficients into spatial orientation tree sets based on the structure of the multi-resolution wavelet decomposition. The algorithm searches for significant coefficient of the wavelet transform along trees rooted at coefficients in the lowest frequency subband and branching into higher frequency subbands along the axes. Table 6.25 shows different rate-distortion values as a function of the granule under study.

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bppb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0.86</td>
<td>42.77</td>
<td>20.74</td>
<td>0.83</td>
</tr>
<tr>
<td>82</td>
<td>1.40</td>
<td>42.03</td>
<td>20.89</td>
<td>0.84</td>
</tr>
<tr>
<td>126</td>
<td>1.34</td>
<td>43.91</td>
<td>19.01</td>
<td>0.82</td>
</tr>
<tr>
<td>151</td>
<td>1.17</td>
<td>44.16</td>
<td>16.36</td>
<td>0.86</td>
</tr>
<tr>
<td>193</td>
<td>1.22</td>
<td>44.25</td>
<td>17.84</td>
<td>0.87</td>
</tr>
</tbody>
</table>
Figure 6.111: Original vs Reconstructed (PSNR = 46.51 dB), Granule 193 (Q1 = 5, Q21 = 6, Q22 = 7)

Figure 6.112: Original vs Reconstructed (PSNR = 57.21 dB), Granule 16 (Q1 = 6, Q21 = 7, Q22 = 8)
Figure 6.113: Original vs Reconstructed (PSNR = 49.33 dB), Granule 82 (Q1 = 6, Q21 = 7, Q22 = 8)

Figure 6.114: Original vs Reconstructed (PSNR = 49.53 dB), Granule 126 (Q1 = 6, Q21 = 7, Q22 = 8)
Figure 6.115: Original vs Reconstructed (PSNR = 48.95 dB), Granule 151 (Q1 = 6, Q21 = 7, Q22 = 8)

Figure 6.116: Original vs Reconstructed (PSNR = 48.66 dB), Granule 193 (Q1 = 6, Q21 = 7, Q22 = 8)
Figures 6.117 through 6.121 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for each of the granules in Table 6.25. Both the compression rate and the distortion are higher when compared to the new compression scheme.

6.3.6 3D SPECK

3D SPECK is the multispectral extension of SPECK, mentioned in Section 2.1.3.3, which encodes 3D volumetric image data by exploiting the dependencies in all dimensions [37]. SPECK has its roots in the ideas developed for SPIHT and therefore has many features and properties similar to SPIHT. SPECK partitions the DWT coefficients into blocks and sorts them by using a quadtree partitioning algorithm [37]. Specifically the algorithm starts by partitioning the image of transformed coefficients into two sets: the set $\mathcal{S}$ which is the root of the pyramid, and the set $\mathcal{I}$ which is everything that is left of the image after taking out the root. Then SPECK sorts the coefficients by testing the significance of the set $\mathcal{S}$ first. The set is declared significant if there is at least one significant coefficient in this set. If $\mathcal{S}$ is found to be significant with respect to the current bit plane, it will be partitioned into four subsets with each subset having approximately one fourth the size of the parent set. Then, SPECK treats each of these four subsets as type $\mathcal{S}$ set and applies the quadtree partitioning recursively to each of them until significant pixels are located. After finishing testing of the type $\mathcal{S}$ sets, the band partitioning is applied to test type $\mathcal{I}$ sets. If the $\mathcal{I}$ set is found to be
Figure 6.118: Original vs Reconstructed (PSNR = 42.03 dB), Granule 82

Figure 6.119: Original vs Reconstructed (PSNR = 43.91 dB), Granule 126
Figure 6.120: Original vs Reconstructed (PSNR = 44.16 dB), Granule 151

Figure 6.121: Original vs Reconstructed (PSNR = 44.25 dB), Granule 193
significant, it will be partitioned into three type $S$ sets and one type $I$ set. By the end of the first pass, many type $S$ sets of varying sizes are generated. In the next pass, SPECK will check the significance against the next bit plane. Table 6.26 shows different rate-distortion values as a function of the granule under study.

Table 6.26: 3D SPECK

<table>
<thead>
<tr>
<th>Granule</th>
<th>Rate (bpb)</th>
<th>PSNR (dB)</th>
<th>Level 2 MPE (%)</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1.13</td>
<td>46.90</td>
<td>14.09</td>
<td>0.87</td>
</tr>
<tr>
<td>82</td>
<td>1.50</td>
<td>45.92</td>
<td>15.42</td>
<td>0.86</td>
</tr>
<tr>
<td>126</td>
<td>1.79</td>
<td>47.60</td>
<td>12.70</td>
<td>0.89</td>
</tr>
<tr>
<td>151</td>
<td>1.58</td>
<td>47.07</td>
<td>13.26</td>
<td>0.88</td>
</tr>
<tr>
<td>193</td>
<td>1.76</td>
<td>46.51</td>
<td>14.52</td>
<td>0.87</td>
</tr>
</tbody>
</table>

Figures 6.122 through 6.126 show spectral bands 0, 150, 300, 700, 1000, 1300, 1700, 2000 comparing the original image against the reconstructed one for each of the granules in Table 6.26. Both the compression rate and the distortion are higher when compared to the new compression scheme.
Figure 6.123: Original vs Reconstructed (PSNR = 45.92 dB), Granule 82

Figure 6.124: Original vs Reconstructed (PSNR = 47.60 dB), Granule 126

213
Figure 6.125: Original vs Reconstructed (PSNR = 47.07 dB), Granule 151

Figure 6.126: Original vs Reconstructed (PSNR = 46.51 dB), Granule 193
6.4 Compression Results

Figure 6.127 shows a comparison between the different rate-distortion curves of the different compression techniques. The plot presents the distortion versus the rate in a logarithm scale to exploit magnitude order difference between the traditional compression techniques and AIRS LP CELP. Specifically the figure shows the data presented in Sections 6.2, 6.3.1, 6.3.2, 6.3.3, 6.3.4, 6.3.5 and 6.3.6. Note that for the AIRS LP CELP scheme only the results obtained when using $5 \times 5$ scanning blocks are presented. The best rate for any distortion happens when using the AIRS LP CELP prediction technique.
Chapter 7

Conclusions and Future Work

7.1 Conclusions

Lossy compression is a solution to the critical problem of storing and transmitting ultraspectral images. Under lossy compression some non essential information, not relevant to the analysis of the image, is discarded improving the compression rate but affecting the overall quality of the image. In this dissertation we focused on AIRS data cubes that include more than 40MB of data spanned over a couple of thousand bands. Specifically we presented a novel architecture that integrates preprocessing and compression stages and provides very efficient lossy compression of AIRS images. On the preprocessing stage the encoder applies some basic reversible process that generates processing side information (unknown to the decoder) beforehand. The decoder reverses the process and applies the same side information to recover the data processed by the encoder. On the compression stage the encoder applies some image compression technique to the original data and based on the desired rate distortion, a result of the distortion measure, it defines the lossy compression parameters as compression side information which are sent to the decoder along with the compressed data resulting from the compression. The decoder uses the compression side information as well as the compressed data to reconstruct the original data cube. Since distortion is introduced in the decompressed data the process is not fully reversible.

During preprocessing the spectral bands are first normalized by readjusting their dynamic range and then reordered in a procedure in which the bands of the image are arranged to increase their similarity measured through the correlation between them. The resulting transformed data cube is then segmented into a combination of variable size spatial blocks that are scanned in a particular fashion based on their spatial location and then concate-
nated into a single unidimensional signal. Since the idea is to minimize high variations to maximize the intrablock correlation during the lossy compression stage, a method of scanning that visits pixels of the block in horizontal and vertical directions is utilized. The output signal is then modeled as an AR process and subjected to linear prediction such that the filter order that guarantees the maximal redundancy removal is selected based on the prediction gain response. The outcome of this procedure are LP coefficients and an error signal that, when encoded, result in a compressed version of the original data cube. The LP coefficients are calculated periodically based on the energy of the prediction error and how it changes in the spatial domain. The encoding of the prediction error can be based either on an open-loop system where the error signal is quantized so it can be later used for synthesis or on a closed-loop system where the error signal is synthesized and fine-tuned as to generate the most accurate reconstruction. The prediction error tends to be approximately a white noise signal that can be further whitened when removing its correlation peaks without significantly increasing the filter order, increasing complexity and coefficients to be transmitted, by cascading long term to short term prediction filters. It can be seen that the long term prediction gain obtained when using three stages is very close to the prediction gain obtained when using a short term prediction filter of equivalent order (four times the scanning block area). Although the long term filters account for reducing most of prediction error induced by the short term filter of lower order, this reduction doesn’t provide a significant improvement that justifies their usage. The most important performance improvement occurs when utilizing short term prediction of filter order equal to the scanning block area as prediction gain is maximized while maintaining the overall compression ratio low.

By taking into account the different individual rates obtained for each of the preprocessing as well as compression stages and combining them into a single equation an overall theoretical compression rate \( r_t \) is obtained for both open-loop and closed-loop (CELP) error encoding. When comparing these rates against the experimental ones \( \hat{r}_t \) obtained by evaluating the data set under study it can be seen that the experimental values are within 25% of the calculated ones. CELP encoding also improves by a 30% the performance of the compression when compared to the open-loop scheme.

In order to mathematically model the rate-distortion an open-loop error encoding scheme is utilized. It is assumed that the error encoding is the only source of distortion and the distortion caused by LP coefficients quantization is masked by the prediction error distortion itself. The distribution of the quantized prediction error is presented as a discrete distribution based on a Laplacian probability density function with parameter \( \lambda \) that accounts for the effect of subband decomposition and band ordering. The overall rate is calculated as the ideal entropy of the source and presented by means of the chain rule as a function of a parameter
related to the quantization level of the prediction error. Similarly the distortion is computed based on the mean square error as a function of the same parameter. The rate-distortion curve is obtained by varying this parameter in order to obtain separate values of both rate and distortion. It can be seen that the experimental and theoretical data points are within 20% of each other. When compared to other six well known multispectral compression techniques, specifically 3D ADPCM, JPEG-7, CALIC, DWT, 3D SPIHT and 3D SPECK, the new ultraspectral compression scheme presented in this dissertation outperforms all of them by providing substantial improvement and delivering both lower compression rates and lower distortion accomplishing rates well below 1 bppb for similar distortion levels.

7.2 Future Work

There are many areas of research related to the current state of the lossy compression architecture presented in this dissertation. The list below details possible major focus areas for future work.

- **Extensions**: Evaluate possible extensions of the architecture to support hyperspectral images, 3D as well as regular video. Compare its rate-distortion pattern against that of those other methods.

- **Optimization**: Analyze the fingerprint of the resource consumption (memory and CPU) of the compression algorithm and research possible improvements.

- **CELP Modeling**: Present a mathematical model to approximate the rate-distortion of a CELP based compression stage and compare it against the real performance of the algorithm.

- **Packetization**: Define an optimal packetization scheme for the output bitstream that will guarantee that most important frames are reliant to network packet loss.

- **Error Correction**: Investigate the overhead incurred by embedding error correction information into the output stream in order to improve reliability and provide packet loss concealment.

- **Encryption**: Address the possible integration of encryption schemes of the output stream and its effect on the overall compression rate.
Appendix A

Rate-Distortion Computation

A.1 Prediction Error Probability

Equation (A.1) below indicates the discrete probability distribution as a function of the quantization interval $i$ of the $j^{th}$ prediction error sample of a given error vector. Note that $f_{X}^{j}(x) = \frac{\lambda_j}{2} e^{-\lambda_j|x|}$ is the probability density function that follows a Laplacian distribution with parameter $\lambda_j$.

$$P (X_j = i) = \int_{Q(i - \frac{1}{2})}^{Q(i + \frac{1}{2})} f_{X}^{j}(x) \, dx$$  \hspace{1cm} (A.1)

Solving (A.1) for $i \neq 0$ the following expression (A.2) is obtained.

$$P (X_j = i) = \frac{\lambda_j}{2} \int_{-\frac{\kappa_j}{2}}^{\frac{\kappa_j}{2}} e^{-\lambda_j Qi} e^{-x} dx$$  \hspace{1cm} (A.2)

which is equivalent to

$$P (X_j = i) = \frac{e^{-\lambda_j Qi}}{2} \left[ e^{\frac{Q \kappa_j}{2}} - e^{-\frac{Q \kappa_j}{2}} \right].$$  \hspace{1cm} (A.3)

Let’s define

$$\kappa_j = \lambda_j Q$$  \hspace{1cm} (A.4)

and

$$\beta_j = \frac{e^{\frac{\kappa_j}{2}} - e^{-\frac{\kappa_j}{2}}}{2} = \sinh \left( \frac{\kappa_j}{2} \right).$$  \hspace{1cm} (A.5)

Replacing (A.4) and (A.5) in (A.3) the following equation (A.6) is obtained.
\[ P(X_j = i) = e^{-\kappa j |i| \beta_j}; i \neq 0 \]  \hspace{1cm} (A.6)

Similarly when \( i = 0 \) the discrete probability is as shown in equation (A.7) below.

\[ P(X_j = 0) = \frac{\lambda_j}{2} \int_{Q^{-\infty}}^{Q^{\infty}} e^{-x} dx \]  \hspace{1cm} (A.7)

This expression can be manipulated as indicated below in equation (A.8).

\[ P(X_j = 0) = 1 - e^{-Q \lambda_j \frac{2}{2}} = 1 - e^{-\frac{\kappa j}{2}} \]  \hspace{1cm} (A.8)

Finally putting (A.6) and (A.8) into a single equation (A.9) the following expression is obtained.

\[ P(X_j = i) = \begin{cases} 
  e^{-\kappa_j |i| \beta_j} & i \neq 0 \\
  1 - e^{-\frac{\kappa_j}{2}} & i = 0 
\end{cases} \]  \hspace{1cm} (A.9)

**A.2 Rate**

The rate \( R_j \) is given by the expectation of the conditional probability as shown by equation (A.10) below (for \( j = 1 \)).

\[ R_j = -E[\log_2 P(X_j)] = - \sum_{i=-\infty}^{\infty} P(X_j = i) \log_2 P(X_j = i) \]  \hspace{1cm} (A.10)

Using equation (A.9) the following expression (A.11) is obtained.

\[ R_j = -\frac{1}{\log_2(2)} \left[ \left( 1 - e^{-\frac{\kappa_j}{2}} \right) \log \left( 1 - e^{-\frac{\kappa_j}{2}} \right) + 2\beta_j \sum_{i=1}^{\infty} e^{-\kappa_j i} \log \left( \beta_j e^{-\kappa_j i} \right) \right] \]  \hspace{1cm} (A.11)

Expanding and using the identities (A.12) and (A.13) below:

\[ \sum_{i=0}^{\infty} x^i = \frac{1}{1 - x} \]  \hspace{1cm} (A.12)
equation (A.10) becomes expression (A.14) that follows.

\[ R_j = -\frac{1}{\log(2)} \left[ \log \left( 1 - e^{-\kappa_j} \right) + e^{-\kappa_j} \log \left( \frac{1 + e^{-\kappa_j}}{2} \right) - \frac{\kappa_j}{2\beta_j} \right] \]  

(A.14)

### A.3 Distortion

The distortion, based on the mean square error and defined as \( D_j \), is given by the equation (A.15) below.

\[ D_j = \sum_{i=\infty}^{i=-\infty} \int_{Q(i+\frac{1}{2})}^{Q(i+\frac{1}{2})} (x - Q_i)^2 f_x(x) \, dx \]  

(A.15)

When replacing with the Laplacian probability density function \( f_x \) and solving the integral the following distortion expression (A.16) is obtained.

\[ D_j = \frac{2e^{-\kappa_j}}{1 - e^{-\kappa_j}} \left[ e^{-\frac{\kappa_j}{2}} \left( \frac{-\kappa_j^2 - 4\kappa_j - 8}{8\lambda_j^2} \right) - e^{-\frac{\kappa_j}{2}} \left( \frac{-\kappa_j^2 + 4\kappa_j - 8}{8\lambda_j^2} \right) \right]. \]  

(A.16)

When applying further manipulation to (A.16) equation (A.17) below is obtained.

\[ D_j = \frac{2}{\lambda_j^2} \left( 2 - \frac{\kappa_j}{\beta_j} \right). \]  

(A.17)

Expression (A.17) can be simplified to become equation (A.18) below.

\[ D_j = \frac{1}{\lambda_j^2} \left( 2 - \frac{\kappa_j}{\beta_j} \right). \]  

(A.18)
Appendix B

Algorithms

This appendix describes some of the algorithms used in the compression architecture presented in this dissertation. Specifically LBG [23] used for codebook generation and Durbin-Levinson [55] used for LP coefficient calculation are detailed.

B.1 LBG

The LBG algorithm is named after Linde, Buzo and Gray but it is also known as generalized Lloyd algorithm which is a generalization of the Lloyd algorithm presented below.

B.1.1 Lloyd Algorithm

Given the codebook size $N$, it is desired to find the input partition cells and codewords such that the average distortion $D = E\{d(x, Q(x))\}$ is minimized, with $x$ being the input random variable or the source with a given probability density function.

It is possible to improve the quantizer by following the two interaction steps indicated below, known as the Lloyd iteration.

i. Given the codebook $Y_m = \{y_{m,i}; i = 1, 2, ..., N\}$, find the optimal partition into quantization cells; that is, use the nearest-neighbor condition to form the nearest-neighbor cells:

$$ R_{m,i} = \{ x : d(x, y_{m,i}) \leq d(x, y_{m,j}) \} $$

for all $j \neq i$. 
ii. Using the centroid condition, find $Y_{m+1}$, the optimal reproduction codewords for the cells just found. Note that the input probability density function must be known in order to compute the centroids.

The Lloyd iteration can be used to improve a quantizer starting from an initial codebook. If the input probability density function is not mathematically tractable, a sample distribution based on empirical observations is used instead. The actual algorithm of Lloyd for quantizer design is stated as follows:

i. Begin with an initial codebook $Y_1$. Set $m = 1$.

ii. Given the codebook $Y_m$ perform the Lloyd iteration to generate the improved codebook $Y_{m+1}$.

iii. Compute the average distortion for $Y_{m+1}$ ($D_{m+1}$). If it has changed by a small enough amount since the last iteration, stop. Otherwise, set $m + 1 \to m$ and go to step ii.

One reasonable stopping criterion is to use the fractional drop in distortion, $(D_m - D_{m+1}) / D_m$. The algorithm stops when the ratio is below a suitable threshold. It can be easily be shown that the algorithm will necessarily produce a sequence of codebooks with monotone non-increasing values of average distortion. If the algorithm converges to a codebook in the sense that further iterations no longer produce any changes in the set of reproduction values, then the resulting codebook must simultaneously satisfy both necessary conditions for optimality.

B.1.2 Generalized Lloyd Algorithm

Consider the distance measure between two $M$-dimensional vectors given by

$$d(x, y) = \|x - y\|^2 = \sum_{m=1}^{M} (x_m - y_m)^2.$$ 

Using this measure, it can be shown that for $x_k \in R_i$, the centroid of $R_i$, denoted by $y_i$, is given by

$$y_i = \frac{1}{N_i} \sum_{x_k \in R_i} x_k$$

with $N_i$ the number of elements in the $i^{th}$ cell. Thus, the centroid of $R_i$ is given by the mean of the elements pertaining to the cell.
Similar to scalar quantization, the solution at convergence depends on the initial codebook. The algorithm converges at local minimums, leading to different levels of performance. A number of random initializations can be employed so that the best codebook (with the minimum distortion sum) is selected as the final solution. An alternative technique is discussed next, which can improve the performance after convergence.

### B.1.3 Stochastic Relaxation

Stochastic relaxation is a method applied to vector quantizer design which is able to avoid poor local minimums. The basic idea is to simultaneously perturb either all encoder or all decoder parameters with an independent white noise process, whose variance starts from a maximum and decreases gradually with the number of iterations. After each perturbation, a Lloyd iteration is performed.

The heuristic justification of the approach is that by adding high levels of noise at the beginning of the training process, the state of the quantizer is essentially randomized. As the noise is reduced, the algorithm finds it more difficult to escape a deep minimum in a single step, but a shallow local minimum will not confine the state. Since the added noise goes to zero, it is much more probable that the final state is in a deep minimum of the total distortion sum.

One proposed perturbation noise variance as a function of the iteration number \( m \) is given by

\[
\sigma_m^2 = \begin{cases} 
A \left(1 - \frac{m-1}{I}\right)^2 & m = 1, \ldots, I \\
0 & m > I 
\end{cases}
\] (B.1)

where \( I \) is a constant defining the number of noisy iterations to run. In general, \( I \) should be large enough so that random perturbation becomes effective. \( A \) is the initial noise variance. According to equation (B.1), noise variance for perturbation is high at the beginning and gradually decreases toward zero. The choice of \( A \) depends on the magnitude of the parameters to be disturbed. One version of the stochastic relaxation algorithm, based on perturbing the training vector set, is outlined below.

i. Begin with an initial codebook \( \mathbf{Y}_1 \). Set \( m = 1 \). The initial codebook can be obtained by randomly selecting some training vectors as codewords.

ii. Perturb the training vector set. Generate noise samples with variance given by equation (B.1). Add these noise samples to the training vector set.
iii. Perform a nearest-neighbor search with the perturbed training vector set to from the cells $R_{m,i}$, $i = 1, ..., N$. With $N$ being the quantizer size, or the number of codewords.

iv. Check for an empty cell: if some of the cells found in step iii are empty, they should be removed to avoid problems with the centroid calculation in the next step. One simple approach is to eliminate the codeword associated with the empty cell and split the cell with the highest number of members into two codewords according to

$$\begin{align*}
    y_e &\leftarrow y_b + u \\
    y_b &\leftarrow y_b - u
\end{align*}$$

where $y_e$ is the codeword associated with the empty cell, $y_b$ is the codeword associated with the biggest cell, and $u$ is a low-norm vector. The vector $u$ can be fixed or generated using a low-variance random number generator. The low-norm requirement is to ensure that the two resultant codewords are not far from the original $y_b$ and hence the new $y_e$ and $y_b$ must take some of the members of the biggest cell, splitting effectively into two nonempty cells. If none of the cells from step iii is empty, the algorithm proceeds to the next step. If some of the cells are empty, a cell-splitting procedure is performed and the algorithm has to go back to step iii again for one nearest-neighbor search. Then step iii is executed again to eliminate empty cells. Unless all cells are nonempty, the algorithm will bound between steps iii and iv.

v. Compute the centroids using the uncorrupted training vectors for the cell just found to obtain the new codebook $Y_{m+1}$.

vi. Compute the distortion sum for $Y_{m+1}$ ($D_{m+1}$). If it has changed by a small enough amount since the last iteration, stop. Otherwise set $m \leftarrow m + 1$ and go to step ii.

The price to pay for using stochastic relaxation is obvious: more training time. In general, the constant $I$ must be high enough so that random perturbation becomes effective. The result at convergence also depends on the initial codebook as well as the noise sequence used for perturbation. Therefore, stochastic relaxation can only be viewed as an alternative training method, and there is no guarantee of achieving global optimality. In practice, the amount of computation incurred in stochastic relaxation might be comparable to running several LBG using different random initializations, and the outcomes of the two approaches might be close. Thus, the advantage of one method over the other is hard to determine or nonexistent.
B.2 Durbin-Levinson

The normal equation as given in (2.6) can be solved by finding the matrix inverse for $R$. In general, inverting a matrix is quite computationally demanding. Fortunately, efficient algorithms are available to solve the equation, which take advantage of the special structure of the correlation matrix. Consider the augmented normal equation of form

$$
\begin{bmatrix}
R_0 & R_1 & \cdots & R_M \\
R_1 & R_0 & \cdots & R_{M-1} \\
\vdots & \ddots & \ddots & \vdots \\
R_M & R_{M-1} & \cdots & R_0
\end{bmatrix}
\begin{bmatrix}
1 \\
a_1 \\
\vdots \\
a_M
\end{bmatrix}
= 
\begin{bmatrix}
J \\
a_1 \\
\vdots \\
0
\end{bmatrix}
$$

(B.2)

which the objective being the solution for the LP coefficients $a_i$, $i = 1, \ldots, M$, given the autocorrelation values $R_l$, $l = 0, 1, \ldots, M$. $J$ represents the minimum mean-squared prediction error or the variance of the input white noise for the AR process synthesizer. In a practical situation, the autocorrelation values are estimated from the signal samples and $J$ is usually unknown; however, the Levinson-Durbin solution is formulated to solve for this quantity as well.

The Levinson-Durbin approach finds the solution to the $M^{th}$ order predictor from that of the $(M-1)^{th}$ order predictor. It is an iterative-recursive process where the solution of the zero-order predictor is first found, which is then used to find the solution of the first-order predictor; this process is repeated one step at a time until the desired order is reached. The algorithm relies on two key properties of the correlation matrix:

1. The correlation matrix of a given size contains as subblocks all the lower order correlation matrices.

2. If

$$
\begin{bmatrix}
R_0 & R_1 & \cdots & R_M \\
R_1 & R_0 & \cdots & R_{M-1} \\
\vdots & \ddots & \ddots & \vdots \\
R_M & R_{M-1} & \cdots & R_0
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
\vdots \\
a_M
\end{bmatrix}
= 
\begin{bmatrix}
b_0 \\
b_1 \\
\vdots \\
b_M
\end{bmatrix}
$$

(B.3)

then

$$
\begin{bmatrix}
R_0 & R_1 & \cdots & R_M \\
R_1 & R_0 & \cdots & R_{M-1} \\
\vdots & \ddots & \ddots & \vdots \\
R_M & R_{M-1} & \cdots & R_0
\end{bmatrix}
\begin{bmatrix}
a_M \\
a_{M-1} \\
\vdots \\
a_0
\end{bmatrix}
= 
\begin{bmatrix}
b_M \\
b_{M-1} \\
\vdots \\
0
\end{bmatrix}
$$

(B.4)

226
that is, the correlation matrix is invariant under interchange of its columns and then its rows. The mentioned properties are direct consequences of the fact that the correlation matrix is Toeplitz. We say that a square matrix is Toeplitz if all the elements on its main diagonal are equal, and if the elements on any other diagonal parallel to the main diagonal are also equal.

We consider the solution to the augmented normal equation starting from zero prediction order. It is shown that the solution for a certain order can be obtained from the lower prediction order results.

### B.2.1 Predictor of Order Zero

In this case we consider the equation

$$R_0 = J_0$$

which is already solved. The above relation states basically that the minimum mean-squared prediction error achievable with a zero-order predictor is given by the autocorrelation of the signal at lag zero, or the variance of the signal itself. For zero order the prediction is always equal to zero; hence, the prediction error is equal to the signal itself. Expanding (B.5) to the next dimension, we have

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} J_0 \\ \Delta_0 \end{bmatrix},$$

which is the two-dimensional version of (B.2) with $a_1 = 0$. Since $a_1 = 0$, the optimal condition cannot be achieved in general, and the term $\Delta_0$ is introduced on the right-hand side to balance the equation. This quantity is found from the equation as

$$\Delta_0 = R_1.$$  

From the property of the correlation matrix, (B.6) is equivalent to

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_0 \\ J_0 \end{bmatrix},$$

Equations (B.6) and (B.8) are used in the next step.
B.2.2 Editor of Order One

We seek to solve

\[
\begin{bmatrix}
  R_0 & R_1 \\
  R_1 & R_0 
\end{bmatrix}
\begin{bmatrix}
  1 \\
  a_1^{(1)} 
\end{bmatrix} = \begin{bmatrix}
  J_1 \\
  0 
\end{bmatrix},
\]

(B.9)

where \(a_1^{(1)}\) is the LP coefficient of the predictor; the superscript denotes the prediction order of one. \(J_1\) represents the minimum mean-squared prediction error achievable using a first-order predictor. Thus, we have two unknowns in equation (B.9): \(a_1^{(1)}\) and \(J_1\). Consider a solution of the form

\[
\begin{bmatrix}
  1 \\
  a_1^{(1)} 
\end{bmatrix} = \begin{bmatrix}
  1 \\
  0 
\end{bmatrix} - k_1 \begin{bmatrix}
  0 \\
  1 
\end{bmatrix},
\]

(B.10)

with \(k_1\) being a constant. Multiplying both sides by the correlation matrix, we have

\[
\begin{bmatrix}
  R_0 & R_1 \\
  R_1 & R_0 
\end{bmatrix}
\begin{bmatrix}
  a_1^{(1)} 
\end{bmatrix} = \begin{bmatrix}
  R_0 & R_1 \\
  R_1 & R_0 
\end{bmatrix}
\begin{bmatrix}
  1 \\
  0 
\end{bmatrix} - k_1 \begin{bmatrix}
  R_0 & R_1 \\
  R_1 & R_0 
\end{bmatrix}
\begin{bmatrix}
  0 \\
  1 
\end{bmatrix},
\]

(B.11)

Substituting (B.6), (B.8), and (B.9) gives

\[
\begin{bmatrix}
  J_1 \\
  0 
\end{bmatrix} = \begin{bmatrix}
  J_0 \\
  \Delta_0 
\end{bmatrix} - k_1 \begin{bmatrix}
  \Delta_0 \\
  J_0 
\end{bmatrix},
\]

(B.12)

Then

\[
k_1 = \frac{\Delta_0}{J_0} = \frac{R_1}{J_0}
\]

(B.13)

where (B.7) is used. The LP coefficient of this predictor is readily found from (B.10) to be

\[
a_1^{(1)} = -k_1
\]

(B.14)

Using (B.12) and (B.13), we find

\[
J_1 = J_0 \left(1 - k_1^2\right).
\]

(B.15)

Thus, the first-order predictor is completely specified. The parameter \(k_1\) is known as the reflection coefficient (RC), representing an alternative form of LP coefficients. Note that \(k_1\) (and therefore \(a_1^{(1)}\) and \(J_1\)) is derived from the results of the previous step: the zero-order
predictor. In a similar manner, we can expand (B.9) to dimension three:

\[
\begin{bmatrix}
R_0 & R_1 & R_2 \\
R_1 & R_0 & R_1 \\
R_2 & R_1 & R_0
\end{bmatrix}
\begin{bmatrix}
1 \\
a_1^{(1)} \\
0
\end{bmatrix}
= 
\begin{bmatrix}
J_1 \\
0 \\
\Delta_1
\end{bmatrix}
\] (B.16)
or

\[
\begin{bmatrix}
R_0 & R_1 & R_2 \\
R_1 & R_0 & R_1 \\
R_2 & R_1 & R_0
\end{bmatrix}
\begin{bmatrix}
0 \\
a_1^{(1)} \\
1
\end{bmatrix}
= 
\begin{bmatrix}
\Delta_1 \\
0 \\
J_1
\end{bmatrix}
\] (B.17)

where \(J_1\) represents the additional term necessary to balance the equation when a first-order predictor is used and \(R_2 \neq 0\). This quantity is solved as

\[\Delta_1 = R_2 + a_1^{(1)} R_1.\] (B.18)

### B.2.3 Predictor of Order Two

We go one step further by solving

\[
\begin{bmatrix}
R_0 & R_1 & R_2 \\
R_1 & R_0 & R_1 \\
R_2 & R_1 & R_0
\end{bmatrix}
\begin{bmatrix}
1 \\
a_1^{(2)} \\
a_2^{(2)}
\end{bmatrix}
= 
\begin{bmatrix}
J_2 \\
0 \\
0
\end{bmatrix}
\] (B.19)

The unknowns in this case are the LP coefficients \(a_1^{(2)}\) and \(a_2^{(2)}\) and the minimum mean-squared prediction error \(J_2\). Consider a solution of the form

\[
\begin{bmatrix}
1 \\
a_1^{(2)} \\
a_2^{(2)}
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
a_1^{(1)} \\
0
\end{bmatrix} - k_2
\begin{bmatrix}
0 \\
a_1^{(1)} \\
1
\end{bmatrix}
\] (B.20)

with \(k_2\) as the RC. Multiplying both sides by the correlation matrix leads to

\[
\begin{bmatrix}
J_2 \\
0 \\
0
\end{bmatrix}
= 
\begin{bmatrix}
J_1 \\
0 \\
\Delta_1
\end{bmatrix}
- k_2
\begin{bmatrix}
\Delta_1 \\
0 \\
J_1
\end{bmatrix}
\] (B.21)
where (B.16), (B.17), and (B.19) are used to derive the above relation. The RC $k_2$ can be found from (B.21) and using (B.18) for $\Delta_1$:

$$k_2 = \frac{1}{J_1} \left( R_2 + a_1^{(1)} R_1 \right).$$

(B.22)

From (B.20), we find

$$a_2^{(2)} = -k_2,$$  

(B.23)

$$a_1^{(2)} = a_1^{(1)} - k_2 a_1^{(1)},$$

(B.24)

Finally, $J_2$ is found from (B.21) and (B.22) as

$$J_2 = J_1 \left( 1 - k_2^2 \right).$$

(B.25)

For the next step, (B.19) is expanded according to

$$\begin{bmatrix}
R_0 & R_1 & R_2 & R_3 \\
R_1 & R_0 & R_1 & R_2 \\
R_2 & R_1 & R_0 & R_1 \\
R_3 & R_2 & R_1 & R_0 \\
\end{bmatrix}
\begin{bmatrix}
1 \\
a_1^{(2)} \\
a_2^{(2)} \\
0 \\
\end{bmatrix}
= 
\begin{bmatrix}
J_2 \\
0 \\
0 \\
\Delta_2 \\
\end{bmatrix}$$

(B.26)

or

$$\begin{bmatrix}
R_0 & R_1 & R_2 & R_3 \\
R_1 & R_0 & R_1 & R_2 \\
R_2 & R_1 & R_0 & R_1 \\
R_3 & R_2 & R_1 & R_0 \\
\end{bmatrix}
\begin{bmatrix}
0 \\
a_1^{(2)} \\
a_2^{(2)} \\
1 \\
\end{bmatrix}
= 
\begin{bmatrix}
\Delta_2 \\
0 \\
0 \\
J_2 \\
\end{bmatrix}.$$

(B.27)

Note that

$$\Delta_2 = R_3 + a_1^{(2)} R_2 + a_2^{(2)} R_1.$$  

(B.28)

**B.2.4 Predictor of Order Three**

In this case, the solution to be considered has the form

$$\begin{bmatrix}
1 \\
a_1^{(3)} \\
a_2^{(3)} \\
a_3^{(3)} \\
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
a_1^{(2)} \\
a_2^{(2)} \\
0 \\
\end{bmatrix}
- k_3 
\begin{bmatrix}
0 \\
a_2^{(2)} \\
a_1^{(2)} \\
1 \\
\end{bmatrix}.$$  

(B.29)
Proceeding in a similar manner, one arrives at the solution

\[ k_3 = \frac{1}{J_2} \left( R_3 + a_1^{(2)} R_2 + a_2^{(2)} R_1 \right), \]  
(B.30)

\[ a_3^{(3)} = -k_3, \]  
(B.31)

\[ a_2^{(3)} = a_2^{(2)} - k_3 a_1^{(2)}, \]  
(B.32)

\[ a_1^{(3)} = a_1^{(2)} - k_3 a_2^{(2)}, \]  
(B.33)

\[ J_3 = J_2 \left( 1 - k_3^2 \right). \]  
(B.34)

The procedure continues until the desired prediction order is reached.

**B.2.5 Summary**

The Levinson-Durbin algorithm is summarized as follows. Inputs to the algorithm are the autocorrelation coefficients \( R_l \), with the LP coefficients and RCs the outputs.

1. Initialization: \( l = 0 \), set

\[ J_0 = R_0. \]

2. Recursion: for \( l = 1, 2, ..., M \)

   (a) Compute the \( l^{th} \) RC

\[ k_l = \frac{1}{J_{l-1}} \left( R_l + \sum_{i=1}^{l-1} a_i^{l-1} R_{l-i} \right). \]  
(B.35)

   (b) Calculate LP coefficients for the \( l^{th} \)-order predictor

\[ a_i^{(l)} = -k_l, \]  
(B.36)

\[ a_i^{(l)} = a_i^{(l-1)} - k_l a_{i-1}^{(l-1)}; i = 1, 2, ..., l - 1. \]  
(B.37)

Stop if \( l = M \).

(c) Compute the minimum mean-squared prediction error associated with the \( l^{th} \)-order solution

\[ J_l = J_{l-1} \left( 1 - k_l^2 \right). \]  
(B.38)

Set \( l \leftarrow l + 1 \); return to step 2a.
The final LP coefficients are

\[ a_i = a_i^{(M)}; \quad i = 1, 2, ..., M. \] (B.39)

Note that in the process of solving the LP coefficients, the set of RCs \((k_i, i = 1, 2, ..., M)\) is also found.

A virtue of the Levinson-Durbin algorithm lies in its computational efficiency. Its use results in a huge saving in the number of operations and storage locations compared to standard methods for matrix inversion. Another benefit of its use is in the set of RCs, which can be used for the verification of the minimum phase property of the resultant prediction-error filter. A system is minimum phase when its poles and zeros are inside the unit circle. Thus, a minimum phase system has a stable and causal inverse. Dependence of the minimum phase condition on RCs is stated in the following theorem.

**Theorem** The prediction-error filter with system function

\[ A(z) = 1 + \sum_{i=1}^{M} a_i z^{-i}, \] (B.40)

where the \(a_i\) are the LP coefficients found by solving the normal equation, is a minimum phase system if and only if the associated RCs \(k_i\) satisfy the condition

\[ |k_i| < 1; \quad i = 1, 2, ..., M. \] (B.41)

The fact that \(A(z)\) represents a minimum phase system implies that the zeros of \(A(z)\) are inside the unit circle of the \(z\)-plane. Thus, the poles of the inverse system \(\frac{1}{A(z)}\) are also inside the unit circle. Hence, the inverse system is guaranteed to be stable if the RCs satisfy condition (B.41). Since the inverse system is used to synthesize the output signal in an LP-based speech coding algorithm, stability is mandatory with all the poles located inside the unit circle. Therefore, by using the Levinson-Durbin algorithm to solve for the LP coefficients, it is straightforward to verify the stability of the resultant synthesis filter by inspecting the RCs. If the magnitudes of the RCs are less than one, the filter is stable.
B.2.6 Conversion of Reflection Coefficients to Linear Prediction Coefficients

As mentioned earlier, the RC represents an alternative form of the LP coefficient. Indeed, a one-to-one correspondence exists between the two sets of parameters. RCs possess several desirable properties, making them the preferred parameters to deal with in many practical situations. Here we consider the problem of finding the LP coefficients given the set of RCs. Consider the set of RCs $k_i; i = 1, ..., M$. It is desired to find the corresponding LP coefficients $a_i$. The problem can be solved directly from the equations in the Levinson-Durbin algorithm, summarized as follows:

For $l = 1, 2, ..., M,$

$$a_l^{(l)} = -k_l,$$  \hspace{1cm} (B.42)

$$a_i^{(l)} = a_i^{(l-1)} - k_i a_{l-i}^{(l-1)}; i = 1, 2, ..., l - 1.$$  \hspace{1cm} (B.43)

At the end of the loop, the desired result is $a_i = a_i^{(M)}$.

B.2.7 Conversion of Linear Prediction Coefficients to Reflection Coefficients

Given the set of LP coefficients $a_i, i = 1, ..., M$, it is desired to find the corresponding RCs $k_i$. This problem can again be solved using the Levinson-Durbin algorithm, working in a reversed fashion. By changing the index in (B.43) to

$$a_{l-i}^{(l)} = a_{l-i}^{(l-1)} - k_i a_i^{(l-1)}$$  \hspace{1cm} (B.44)

and substituting (B.44) in (B.43) to eliminate $a_{l-i}^{(l-1)}$ leads to

$$a_i^{(l)} = a_i^{(l-1)} - k_i a_{l-i}^{(l-1)} - k_i^2 a_i^{(l-1)},$$

or

$$a_i^{(l-1)} = \frac{a_i^{(l)} + k_i a_{l-i}^{(l)}}{1 - k_i^2}.$$  \hspace{1cm} (B.45)

The above equation is used to find the RCs based on the following loop, with $a_i^{(M)} = a_i$:

For $l = M, M - 1, ..., 1,$
\[ k_l = -a_i^{(l)}, \quad (B.46) \]

\[ a_i^{(l-1)} = \frac{a_i^{(l)} + k_l a_{l-i}^{(l)}}{1 - k_l^2}; \ i = 1, 2, \ldots, l - 1. \quad (B.47) \]
Bibliography


[27] Q. Du, C. Chang, “Linear mixture analysis-based compression for hyperspectral image analysis”, IEEE transactions on geoscience and remote sensing, 2004


238


