Entwicklungen verteilter Bildercodierungsmethoden basierend auf LDPC

Vorgelegt von
Diplom-Ingenieur
Georges Haboub

aus Berlin

Von der Fakultät IV – Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuss
Vorsitzender: Prof. Dr.-Ing. R. Orglmeister
1. Gutachter: Prof. Dr.-Ing. T. Sikora
2. Gutachter: Prof. Dr.-Ing. P. Eisert

Tag der wissenschaftlichen Aussprache: 01.07.2011

Berlin 2011
D 83

Zusammenfassung

Diese Arbeit beschäftigt sich mit der verteilten Codierung von Einzelbildern (engl: distributed image coding = DIC). Die DIC unterscheidet sich insofern von der traditionellen Bildercodierung, als dass der Codierungsprozess für einen Teil des Bildes keine räumliche Prädiktion vewendet. Stattdessen wird eine typische Kanalcodierung auf den Transformationsparametern dieses Teils des Bildes durchgeführt. Wir bezeichnen diesen Teil als Wyner-Ziv-Bild, den zweiten Teil als Schlüssel-Bild. Das herkömmlich codierte Schlüssel-Bild wird bei der Decodierung verwendet um eine Schätzung des Wyner-Ziv-Bildes durchzuführen, welches mit Hilfe des Kanalcodes verfeinert wird. Der Vorteil dieses Ansatzes gegenüber bekannten Bildcodierungsverfahren wie JPEG, JPEG2000 oder auch H.264 Intra-Codierung besteht vor allem in der Verschiebung der Codierungskomplexität für das Wyner-Ziv-Bild vom Encoder in den Decoder.

Der vewendete Ansatz basiert auf der Theorie der Wyner-Ziv- und der Slepian-Wolf-Codierung, die eng miteinander verbunden sind. Es werden beide Codierungsarten angewandt, um das Wyner-Ziv-Bild zu codieren. Ausgehend von der Theorie der verteilten Quellencodierung werden Wyner-Ziv-Bild und das Schlüssel-Bild unabhängig voneinander codiert.

Die verwendete Wyner-Ziv-Codierung in dieser Arbeit führt zunächst die räumliche Dekorrelation des Wyner-Ziv-Bildes mittels der ganzzahligen Diskreten Cosinus-Transformation (engl: integer discrete cosine transform = DCT) durch, anschließend wird die Slepian-Wolf-Codierung (Kanalcodierung) angewandt, um die unkorrelierte Information zu codieren. Hierbei wird die Methode der "ASWC (asymmetrische Slepian-Wolf-Codierung) mit der Paritätsmethode" eingesetzt. Es werden also ausschließlich Paritätsbits erzeugt und über einen Kanal versendet. Zur Konstruktion der Paritätsbits wird der Niedrigdichten-Paritäs-Prüfcode (engl: low density parity check code = LDPC) benutzt.

In dieser Arbeit werden insgesamt drei praktische DIC-Verfahren mit der Intra-Codierung H.264 verglichen. Diese Verfahren unterscheiden sich vor allem in der räumlichen Aufteilung eines zu codierenden Bildes in das Schlüssel-Bild und das Wyner-Ziv-Bild, bzw. in der Schätzung des Wyner-Ziv-Bildes aus dem Schlüssel-Bild. Eine gute Schätzung trägt zur Güte des Verfahrens bei indem bei gleicher Qualität die Zahl der zu sendenden Paritätsbits geringer ausfällt. Zur Auswertung werden objektiv messbare und subjektiv wahrnehmbare Bewertungen herangezogen. Der Kompressionsfaktor und die korrigierte Bit-Rate der Bitebenen des Wyner-Ziv-Bildes, Codierungskomplexität und Decodierungskomplexität werden ebenfalls besprochen. Bei subjektiv gleicher Qualität wurden in dieser Arbeit Einsparungen in der Komplexität von bis zu 42% im Encoder – verglichen mit dem H.264 Intra-Encoder – erreicht.

Abstract

This work focuses on distributed image coding (DIC). The DIC is different from traditional Image coding approaches in a way that the coding process does not use intra-prediction for a part of the image.

Instead of intra-prediction it uses channel coding on the transformation parameters for that particular part of an image. We refer to this image part as Wyner-Ziv-Image, the second part is called the Key-Image. The traditionally coded Key-Image will be used by the decoder to predict the Wyner-Ziv-Image. This predicted Wyner-Ziv-Image will be improved by using channel coding methods. The main advantage of this approach compared to well known image coding approaches, e.g. JPEG, JPEG2000 or H.264 Intra-Coding, is that coding complexity of Wyner-Ziv-Image is shifted to the decoder.

Our approach exploits the theory of the Wyner-Ziv- and Slepian-Wolf-Coding which relate to each other. Both of the coding approaches (Wyner-Ziv- and Slepian-Wolf-Coding) are used to code the Wyner-Ziv-Image. Based on the theory of distributed source coding we will code the Wyner-Ziv-Image and Key-Image independently.

The used Wyner-Ziv-Coding in our work starts by de-correlate the Wyner-Ziv-Image using integer cosine transformation (DCT), afterword we use the Slepian-Wolf-Coding (channel coding) to code the de-correlated information. Hereby the "ASWC (asymmetric Slepian-Wolf-Coding) with Parity approach" is used. At the end, the parity bits are sent through that channel. To generate the parity bit we use LDPC.

In our work we compare three realistic distributed image coding approaches against Intra-Coding H.264. These approaches are different. Firstly, they differ in the partition process of the image to encode into two parts (Wyner-Ziv-Image and Key-Image). Secondly, we assess different ways of predicting the Wyner-Ziv-Image from the Key-Image. For good predictions the needed parity bits are reduced to achieve the same image quality as in Intra-Coding H.264.

To evaluate the results we use objective and subjective perceived measures. We also analyse the compression factors and the corrected bit rate of the bit planes for the Wyner-Ziv-Image. Moreover, we also evaluate the complexity of the coding and the decoding.

In our work we reduce the coding complexity by maximum of 42% compared to the Intra-Coding H.264 to achieve the same subjective quality of the image.

Erklärung

Hiermit versichere ich an Eides statt, die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als der von mir angegebenen Quellen angefertigt zu haben. Alle aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche gekennzeichnet. Die Arbeit wurde noch keiner Prüfungsbehörde in gleicher oder ähnlicher Form vorgelegt.

Berlin, den 13. April 2011

(Georges Haboub)

Danksagung

Beginnen möchte ich mit den Personen, denen meine Dissertation gewidmet ist. Dies ist meine Familie. Ohne die grenzenlose und uneingeschränkte Unterstützung meiner Eltern wäre ich nicht in der Position in der ich jetzt glücklicherweise bin. Sie waren immer für mich da und haben gemeinsam mit meinem Bruder meine innere Ausgeglichenheit und Stärke aufgebaut, die ich während meiner Dissertationsarbeit dringend gebraucht habe.

Nicht weniger zu danken möchte ich meinem Doktorvater, Herr Prof. Dr.-Ing. Thomas Sikora für die wohlwollende Begleitung meines Promotionsverfahrens, wie all meinen Kollegen. Ganz herzlich danke ich Dr.-Ing Matthias Kunter für seine Ratschläge und Diskussionen sowie für seine Freundschaft. Für die fachlichen Hinweise und die konstruktive Kritik bedanke ich mich bei meinem Zweitgutachter Herrn Prof. Dr.-Ing. Peter Eisert.

Mein letzter und größter Dank gilt jedoch meiner Frau. Gemeinsam mit meinem Sohn Rimon war sie zu jeder Zeit für mich da und hat mich in all meinen anstrengenden Tagen begleitet und mich jederzeit mit voller Kraft unterstützt. Ich möchte mich hiermit für ihre unausschöpfbare Liebe und für Ihre Kraft in diesem Rahmen ganz herzlich bei ihr bedanken. Ich liebe dich.

Dipl.-Ing. Georges Haboub

Inhaltsverzeichnis

1 Einführung ... 1

1.1 Aufbau der Arbeit ... 2

2 Information zu den theoretischen Hintergründen ... 3

2.1 Die Entropie, ein stochastischer Prozess ... 3

2.2 Theorem der fehlerfreien Quellencodierung und die Rateverzerrung -Funktion ... 5

2.2.1 Rateverzerrung-Funktion eines Bildes ... 8

2.3 Die Kanalkapazität und das Theorem der Kanalcodierung ... 9

3 Überblick über die verteilte Quellencodierung ... 10

3.1 Der virtuelle Kanal in der verteilten Quellencodierung ... 11

3.2 Die binäre symmetrische virtuelle Kanal bei der verteilten Quellencodierung ... 12

3.3 Verlustfreie Slepian-Wolf-Quellencodierung ... 13

3.3.1 Asymmetrische Slepian-Wolf-Codierung (ASWC) ... 14

3.3.2 Nicht-asymmetrische Slepian-Wolf-Codierung ... 30

3.4 Wyner-Ziv-Codierung ... 35

3.4.1 Wyner-Ziv-Coderuing für den binären symmetrischen Fall ... 36

3.4.2 Wyner-Ziv-Codierung für den Quadratischen Gaußschen Fall ... 39

3.4.3 Praktische Wyner-Ziv-Codierung mittels verschachtelt skalierter Quantisierung ... 40

3.5 Simulationsergebnisse für die praktische SW/WZ bei Nutzung von Kanalcodierung ... 44

4 Niedrigdichten Paritätsprüfcode (Low density parity check code=LDPC) ... 46

4.1 Einleitung ... 46

4.2 Reguläre Konstruktion eines LDPC-Codes ... 47

4.3 Der Codierungsalgorithmus ... 49

4.4 Fortschritt bei der Komplexitätsreduktion der LDPC-Encoder ... 49

4.5 Die Tanner-Graphen ... 53

4.6 Weiche LDPC-Decodierung ... 54

4.6.1 Das Prinzip des Decodierungsalgorithmus ... 54

4.6.2 Der Decodierungsalgorithmus im Detail ... 55

4.6.3 Die Zusammenfassung des Decodierungsalgorithmus für einen binären LDPC ... 65

4.7 Der error-floor des LDPC ... 67

4.8 Harte LDPC-Decodierung ... 69

4.9 Der Summen-Produkt-Algorithmus (SPA) ... 73

4.10 Der irreguläre LDPC-Code ... 83

4.11 Density evolution ... 85

4.12 ASWG Paritätsmethoden mit LDPC bei Nutzung der Bitebenen-Faktorisierung ... 97

5 State of the Art verteilte Bild- und Videocodierung ... 104

6 Implementierung und Bewertung der Methoden zur verteilten Bildcodierung ... 107

6.1 Annahmen für die Simulationen, Maße für Leistung und die Rate ... 108

6.2 Mehode I:Verteilte Bildercodierung mit Intra-Vorhersage-Modi ... 114

6.2.1 Der Encoder ... 114

6.2.2 Der Decoder ... 121

6.2.3 Bewertung der Ergebnisse ... 125

6.2.4 Zusammenfassung ... 151

6.3 Methode II: Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung ... 152

6.3.1 Maximal glatte Bild-Wiederherstellung mit der Transformationscodierung (MGBW) ... 152

6.3.2 Codierung und Decodierung ... 155

6.3.3 Bewertung der Ergebnisse ... 156

6.3.4 Zusammenfassung ... 175

6.4 Methode III:Verteilte Bildercodierung mit linearer Pixel-Vorhersage ... 176

6.4.1 Codierung und Decodierung ... 176

6.4.2 Bewertung der Ergebnisse ... 178

6.4.3 Zusammenfassung ... 197

6.5 Vergleich der Arbeit mit anderen Papers der verteilten Bildcodierung ... 198

7 Zusammenfassung und Ausblick ... 201

7.1 Zusammenfassung ... 201

7.2 Ausblick ... 202

8 Anhänge ... 205

8.1 Visueller Vergleich zwischen Methode I und H.264 Intra-Coding ... 205

8.2 Visueller Vergleich zwischen Methode II und H.264 Intra-Coding ... 212

8.3 Original Bilder und Visueller Vergleich zwischen Methode III und Intra-Coding ... 219

8.3.1 Original WZ-Bilder und Schlüßel-Bilder ... 219

8.3.2 Visueller Vergleich zwischen Methode III und H.264 Intra-Coding ... 220

8.4 Visueller Vergleich zwischen Methode I, Methode II, Methode III ... 228

Tabellenverzeichnis

Tabelle 4.1: Grenzwerte für verschiedene irreguläre LDPC mit Rate 0.5 bei binären AWGN Kanal. 85
Tab. 4.2: Grenzwerte verschiedener LDPC-Codes für den binären AWGN-Kanal. 96
Tab. 4.3: Grenzwerte verschiedener LDPC-Codes für den BSC. 96
Tab. 6.1: Maß spektraler Konstantz der Bilder 112
Tab. 6.2: Grenzwerte verschiedener unsere LDPC-für den BSC. 113
Tab. 6.3: Quantisierung Schrittweite in H.264 codec. 118
Tab. 6.4: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode I. 140
Tab. 6.5: Die Zeit der Codierung in Sekunden für die Bilder *Haus, Lena* in Methode I im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter. 146
Tab. 6.6: Die Zeit der Codierung in Sekunden für die Bilder *Barbara, Baboon* in Methode I im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter. 146
Tab. 6.7: Zeit der Decodierung in Sekunde für die Bilder *Haus, Lena* in Methode I im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 147
Tab. 6.8: Zeit der Decodierung in Sekunde für die Bilder *Barbara, Baboon* in Methode I im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 148
Tab. 6.9: Vergleich der Anzahl der korrigierten Bitebenen der Koeffizienten zwischen Methode I und Methode II, wenn der Quantisierungsparameter des Schlüssel-Bildes qp=0 ist. 163
Tab. 6.10: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode II. 168
Tab. 6.11: Die Zeit der Codierung in Sekunden für die Bilder *Haus, Lena* in Methode II im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter. 171
Tab. 6.12: Die Zeit der Codierung in Sekunden für die Bilder *Barbara, Baboon* in Methode II im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter. 172
Tab. 6.13: Zeit der Decodierung in Sekunde für die Bilder *Haus, Lena* in Methode II im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 173
Tab. 6.14: Zeit der Decodierung in Sekunde für die Bilder *Barbara, Baboon* in Methode II im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 173
Tab. 6.15: Vergleich der Anzahl der korrigierten Bitebenen der Koeffizienten zwischen Methode I, Methode II und Methode III, wenn der Quantisierungsparameter des Schlüssel-Bildes qp=0 ist. 184
Tab. 6.16: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode III. 189
Tab. 6.17: Die Zeit der Codierung in Sekunden für die Bilder *Haus, Lena* in Methode III im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter. 193
Tab. 6.18: Die Zeit der Codierung in Sekunden für die Bilder *Barbara, Baboon* in Methode III im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter 193
Tab. 6.19: Zeit der Decodierung in Sekunde für die Bilder *Haus, Lena* in Methode III im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 194
Tab. 6.20: Zeit der Decodierung in Sekunde für die Bilder *Barbara, Baboon* in Methode III im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter. 195

Abbildungsverzeichnis

Abb. 2.1: Stochastischer Prozesses besteht aus 4 Zufallsvariablen. 3
Abb. 2.2: Übertragungssystem. 6
Abb. 2.3: Allgemeine Rate-Verzerrung-Kurve. 8
Abb. 3.1: Bildliche Darstellung von verteilter Quellencodierung für mehrere abhängige Sensoren. 11
Abb. 3.2: Das virtuelle Kanalmodell I zwischen Quelle X und Zusatzinformation Y. 12
Abb. 3.3: Das virtuelle Kanalmodell II zwischen Quelle X und Zusatzinformation Y. 12
Abb. 3.4: Das binäre symmetrische Kanalmodell (BSC) für unabhängige identisch verteilte binäre Quellen X und Y. 13
Abb. 3.5: Gemeinsame Codierung und Decodierung von X und Y. 14
Abb. 3.6: Verteilte Quellencodierung statistisch abhängiger i.i.d. diskreter Zufallssequenzen X und Y. 14
Abb. 3.7: Bereich der erreichbaren Raten von WZ für statistisch i.i.d. diskrete Zufallssequenzen X und Y. 15
Abb. 3.8: Verteilte Quellencodierung für mehrere Quellen. 18
Abb. 3.9: ASWC mit der Syndrom-Methode 24
Abb. 3.10: Paritätsmethode für anpassungsfähigen code. 28
Abb. 3.11: Bereich der erreichbaren Raten von WZ für statistisch i.i.d. diskrete Zufallssequenzen X und Y, wenn die Korrelation zwischen den Qellen gerniger wird. 30
Abb. 3.12: Blockdiagramm einer allgemeinen Wyner-Ziv Codierung. 37
Abb. 3.13: Simulierte Leistung des verschachtelten Schemas aus [LXG03] für die binäre Wyner-Ziv-Codierung bei einer Korrelation P = 0.27. Die Zeitteillinie (engl: Time-Scharing) zwischen dem Null-Raten-Punkt (P, 0) und dem Slepian-Wolf-Punkt (0, H(P)) wird auch gezeigt.Quelle: [LXG03]. 39
Abb. 3.14: Blockdiagramm der praktischen Wyner-Ziv-Codierung. 41
Abb. 3.15: 1-D verschatelte Lattice gleichförmige Quantisierung mit vier Teilmengen (engl:bins) für das Quadratische Gaußsche Wyner-Ziv-Problem, wobei die Zusatzinformation Y nur am Decoder zur Verfügung steht. (a) "nur gute" Verzerrung. (b) "gute" und "schlechte" Verzerrung. 42
Abb. 3.16: Die theoretischen Grenzen für WZ/SW-Codierung im Vergleich mit regulären und irregulären LDPC- und Turbo-Codes. Quelle: [LXG02]. 45
Abb. 4.1: Neue Form der Matrix H zur Konstruktion einer einfachen LDPC-Codierung. 50
Abb. 4.2: Tanner-Graph für einen Beispiel-Matrix H. 54
Abb. 4.3: Nachrichtenaktualisierung beim iterativen. 56
Abb. 4.4: Teilmenge des Tanner-Graphen mit Zirkel ($c1, z1, c2, z5$), mit $c1$ als Wurzel. 58
Abb. 4.5: Das Trellis-Diagrammfür die Ermittlung von $rmn(x)$ in GF(3). 62
Abb. 4.6: Decodierungsleistung für LDPC-Codes mit den Raten R=1/2 und R=1/3.Quelle: [Moo05]. 66
Abb. 4.7: Durchschnittliche Anzahl von Decodierungsiterationen für LDPC mit R=1/3 N=15000 und R=1/2 N=20000. Quelle: [Moo05]. 66

Abb. 4.8: Leistung der Bitfehlerrate und Frame-Fehlerrate eines 0.9(4550, 4096) quasi-zirkularen IRA-Codes mit SPA und min-sum-Decodierung. Quelle: [WER09]. 68
Abb. 4.9: Beispiel für ein trapping set (3,1). 68
Abb. 4.10: Decodierungsstruktur des log-likelihood Decodierungsalgorithmus mit der Funktion Φ(x). 83
Abb. 4.11: Die PDF von mv wenn die übertragene Nachricht ein Einsen-Code-Wort ist bei Eb/N0 =1.72dB (final), und Eb/N0 =1.764 dB (für verschiedene Iterations). Quelle: [Moo05]. 95
Abb. 4.12: Der Mittelwert der PDF $\mathcal{P}$kv als Funktion der Iteration k bei verschiedenen Werten von Eb/N0, wenn die übertragene Nachricht ein Einsen-Code-Wort ist. Quelle: [Moo05]. 95
Abb. 4.13: Die Laplacesche PDF der Differenz zwischen Quelle X und Zusatzinformation Y. 101
Abb. 4.14: Finden der bedingten Wahrscheinlichkeit von $Qb = 0, 1$ bei gegebenem bedingtem Wahrscheinlichkeitsdichte-Modell P(Q/Y) und gegebenen Werten aller zuvor decodierten Bitebenen $Qb + 1, \dots, QL$. 103
Abb. 6.1: Die Originalbilder *Haus*, *Lena*, *Barbara*, *Baboon* von links nach rechts und von oben nach unten. 110
Abb. 6.2: Die logarithmischen Leistungsdichtespektren der Bilder *Haus*, *Lena*, *Barbara*, *Baboon* (von links nach rechts und von oben nach unten). Die μ-Achse und v-Achse stellen die Frequenz entlang der waagerechten und senkrechten Richtung des Bildes dar. Auf der A-Achse sind die logarithmischen Leistungsdichtespektren der Bilder aufgetragen. 111
Abb. 6.3: Blockdiagramm der verteilten Bildercodierung mit Intra-Vorhersage-Modi. 115
Abb. 6.4: Das Schachbrett-Muster nach dem das Bild in WZ-Bild (weiße Blöcke) und Schlüssel-Bild (schwarze Blöcke) aufgeteilt wird. 115
Abb. 6 .5: Der 4×4 DCT Koeffizienten-Block, die Zahlen 1:16 entsprechen den Positionen der Koeffizienten von DC bis AC15. 117
Abb. 6.6: Luminanz-Vorhersage-Modi bei H.264. die Pixel A,B,C,D gehören zum oberen Block, die Pixel I,J,K,L zum linken Block, die Pixel M zum oberen linken Block, die Pixel E,F,G,H gehören zum oberen rechten Block im Verhältnis zum betrachteten Block. Der Pfeil zeigt die Richtung der Luminanzänderung beim Vorhersage-Modus an. 121
Abb. 6.7: Originalbild (*Barbara*), WZ-Bild und Schlüssel-Bild bei Schachbrett-Aufteilung. 127
Abb. 6.8: Die Original-WZ-Bilder links und die Original-Schlüssel-Bilder rechts, die Bilder wurden mit Methode I (Verteilte Bildercodierung mit Intra-Vorhersage-Modi) codiert. 127
Abb. 6.9: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit Intra-Vorhersage-Modi für die Bilder *Haus* (oben) und *Lena* (unten). Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 9 sind qp=(0,5,10,15,30,35,40,45,50). Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 10 sind qp=(0,5,10,15,20,25,30,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 129
Abb. 6.10: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit Intra-Vorhersage-Modi für die Bilder *Barbara* (oben) und *Baboon* (unten). Der

Quantisierungsparameter des Schlüssel-Bildes von Kurve1 bis Kurve 10 sind qp=(0,5,10,15,20,25,30,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 130

Abb. 6.11: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode I und der H.264 Intra-Codierung für die Bilder *Haus* (oben) und *Lena* (unten). 132

Abb. 6.12: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode I und der H.264 Intra-Codierung für die Bilder *Barbara* (oben) und *Baboon* (unten) 133

Abb. 6.13: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild Haus mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes ist qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 136

Abb. 6.14: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes ist qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0) 137

Abb. 6.15: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 138

Abb. 6.16: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0) 139

Abb. 6.17: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *DC, AC1* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet. 142

Abb. 6.18: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC2, AC3* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet. 143

Abb. 6.19: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC13* , *AC14* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet. 144

Abb. 6.20: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn die Qualität der Zusatzinformation sehr niedrig ist (der Quantisierungsparameter des Schlüssel-Bildes ist qp=50), PSNR dieses Bildes ist 22.18 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp. 149

Abb. 6.21: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn die Qualität der Zusatzinformation sehr niedrig ist (der Quantisierungsparameter des Schlüssel-Bildes ist qp=50), PSNR dieses Bildes ist 23.88 dB und die Rate 1 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.035 dB und die Rate 1.001 bpp. 149

Abb. 6.22: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn Der Quantisierungsparameter des Schlüssel-Bildes qp=40 ist, PSNR dieses Bildes ist 26.46 dB und die Rate 0.35 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=39), PSNR diese Bildes ist 28.164 dB und die Rate 0.362 bpp. 150

Abb. 6.23: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn Der Quantisierungsparameter des schlüssel-Bildes qp=40 ist, PSNR dieses Bildes ist 29.04 dB und die Rate 1.09 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp. 151

Abb. 6.24: Darstellung der Glättungsbedingungen: Ein Pfeil zwischen zwei Samples bedeutet, dass die Glätte-Differenz zwischen diesen zwei Samples im Maß für die Glätte (siehe Gleichung 6.11) vorkommt. Die dunkelgrauen Samples sind die äußeren Grenz-Samples a) Die Glättungsbedingung wird nur an der Grenze angewandt. b) Die Glättungsbedingung wird an jedem Sample in Richtung der nächsten Grenze angewandt. 154

Abb. 6.25: Das Block-Diagramm der verteilten Bildercodierung mit maximal glatter Bild-Wiederherstellung. 156

Abb. 6.26: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Haus*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 7 sind qp=(0,5,10,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 157

Abb. 6.27: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung,mit Intra-Vorhersage-Modi für Bild *Lena*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 9 sind

qp=(0,5,10,15,20,25,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 157

Abb. 6.28: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Barbara*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 sind qp=(0,10,20,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 158

Abb. 6.29: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Baboon*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,30,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 158

Abb. 6.30: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode II und der H.264 Intra-Codierung für die Bilder *Haus* (oben) und *Lena* (unten). 161

Abb. 6.31: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode II und der H.264 Intra-Codierung für die Bilder *Barbara* (oben) und *Baboon* (unten) 162

Abb. 6.32: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Haus* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 164

Abb. 6.33: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 165

Abb. 6.34: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 166

Abb. 6.35: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 167

Abb. 6.36: Das Histogramm der Differez zwischen dem Koeffizienten-Band *DC* des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes sowie die PDF zwischen dem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode I und II berechnet. 169

Abb. 6.37: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC1,AC2* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode I und II berechnet. 170

Abb. 6.38: Das linke Bild ist das Vorhergesagte Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp = 40, PSNR dieses Bildes ist 24.18 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp.174

Abb. 6.39: Das linke Bild ist das Vorhergesagte Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp =40, PSNR dieses Bildes ist 26.96 dB und die Rate 1.14 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp. 175

Abb. 6.40: Blockdiagramm der verteilten Bildercodierung mit linearer Pixel-Vorhersage. 177

Abb. 6.41: Das Spalten-Muster nach dem das Bild in WZ-Bild (weiße Spalten) und Schlüssel-Bild (schwarze Spalten) aufgeteilt wird. 177

Abb. 6.42: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Haus*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 qp=(0,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 179

Abb. 6.43: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Lena*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 179

Abb. 6.44: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Barbara*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 sind qp=(0,10,20,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker

stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 180

Abb. 6.45: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Boaboon*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,30,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15. 180

Abb. 6.46: Originalbild (*Barbara*), WZ-Bild und Schlüssel-Bild bei Spalten-Aufteilung. 181

Abb. 6.47: Vergleich zwischen Maximum optimum Coding der Methode III und H.264 Intra-Codierung und Methode I und Mthode II für die Bilder *Haus* (oben) und *Lena* (unten). 182

Abb. 6.48: Vergleich zwischen Maximum optimum Coding der Methode III und H.264 Intra-Codierung und Methode I und Mthode II für die Bilder *Barbara* (oben) und *Baboon* (unten). 183

Abb.6.49: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Haus* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 185

Abb.6.50: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 186

Abb.6.51: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0). 187

Abb. 6.52: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0) 188

Abb.6.53: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *DC,AC1* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode III und II berechnet. 191

Abb.6.54: Das Histogramm der Differez zwischen dem Koeffizienten-Band *AC2* des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes sowie die PDF zwischen dem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode III und II berechnet. 192

Abb. 6.55: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=40, PSNR dieses Bildes ist 24.23 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp. 196

Abb. 6.56: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung Verteilte Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=40, PSNR dieses Bildes ist 26.23 dB und die Rate 1.06 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp. 196

Abb. 8.1: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 26.76 dB und die Rate 0.18 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 35.859dB und die Rate 0.183bpp. 205

Abb. 8.2: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 28.05 dB und die Rate 0.73bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp. 205

Abb. 8.3: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 35.83dB und die Rate 0.36 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=27), PSNR diese Bildes ist 40.02 dB und die Rate 0.373 bpp. 206

Abb. 8.4: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 38.55 dB und die Rate 1.03 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=18), PSNR diese Bildes ist 45.52 dB und die Rate 1.013 bpp. 206

Abb. 8.5: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 28.03 dB und die Rate 0.31 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=37), PSNR diese Bildes ist 30.619 dB und die Rate 0.329 bpp. 207

Abb. 8.6: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 30.15 dB und die Rate 0.86 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 36.915 dB und die Rate 0.829 bpp. 207

Abb. 8.7: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi, Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 31.7 dB und die Rate 0.83 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 36.915 dB und die Rate 0.829 bpp. 208

Abb. 8.8: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 36.23 dB und die Rate 1.32 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=23), PSNR diese Bildes ist 40.612 dB und die Rate 1.277 bpp. 208

Abb. 8.9: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 29.53 dB und die Rate 0.96 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.035 dB und die Rate 1.001 bpp. 209

Abb. 8.10: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 35.41dB und die Rate 1.67 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=22), PSNR diese Bildes ist 42.092 dB und die Rate 1.715 bpp. 209

Abb. 8.11: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 21.21 dB und die Rate 0.18 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=44), PSNR diese Bildes ist 22.722 dB und die Rate 0.167 bpp. 210

Abb. 8.12: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 22.65 dB und die Rate 1.02 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 29.52 dB und die Rate 1.009 bpp. 210

Abb. 8.13: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 24.25 dB und die Rate 0.38 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=40), PSNR diese Bildes ist 24.976 dB und die Rate 0.396 bpp. 211

Abb. 8.14: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 26.34 dB und die Rate 1.22 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=32), PSNR diese Bildes ist 31.104 dB und die Rate 1.259 bpp. 211

Abb. 8.15: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 29.72 dB und die Rate 0.08 bpp. Das rechte Bild wurde komplett mit H.264codiert (qp=43), PSNR diese Bildes ist 30.86 dB und die Rate 0.08 bpp. 212

Abb. 8.16: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 33.18 dB und die Rate 0.75 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp. 212

Abb. 8.17: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 32.54 dB und die Rate 0.34 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 39.506 dB und die Rate 0.34 bpp. 213

Abb. 8.18: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 39.16 dB und die Rate 1.08 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=18), PSNR diese Bildes ist 45.520 dB und die Rate 1.013 bpp. 213

Abb. 8.19: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 25.62dB und die Rate 0.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 26.727 dB und die Rate 0.159 bpp. 214

Abb. 8.20: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 29.04 dB und die Rate 0.76 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.073 dB und die Rate 0.746 bpp. 214

Abb. 8.21: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 27.42 dB und die Rate 0.44 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 32.607 dB und die Rate 0.451 bpp. 215

Abb. 8.22: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 33.48 dB und die Rate 1.04 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=26), PSNR diese Bildes ist 38.373 dB und die Rate 0.988 bpp. 215

Abb. 8.23: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=0. PSNR dieses Bildes ist 26.08 dB und die Rate 2.89 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=13), PSNR diese Bildes ist 49.923dB und die Rate 2.936 bpp. 216

Abb. 8.24: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der

Quantisierungsparameter des Schlüssel-Bildes qp=0. PSNR dieses Bildes ist 30.89 dB und die Rate 3.82 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=8), PSNR diese Bildes ist 54.117 dB und die Rate 3.76 bpp. 216

Abb. 8.25: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 20.59 dB und die Rate 0.03 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=51), PSNR diese Bildes ist 20.778 dB und die Rate 0.034 bpp. 217

Abb. 8.26: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 22.1 dB und die Rate 0.80 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=36), PSNR diese Bildes ist 27.82 dB und die Rate 0.765 bpp. 217

Abb. 8.27: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 24.66 dB und die Rate 0.86 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=35), PSNR diese Bildes ist 28.605 dB und die Rate 0.874 bpp 218

Abb. 8.28: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 28.66 dB und die Rate 1.71 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 33.699 dB und die Rate 1.7 bpp. 218

Abb. 8.29: Die Original-WZ-Bilder rechts und die Original-Schlüssel-Bilder links, in Methode III (verteilte Bildercodierung mit linearer Pixel-Vorhersage)..... 219

Abb. 8.30: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=45. PSNR dieses Bildes ist 28.23 dB und die Rate 0.04 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=49), PSNR diese Bildes ist 26.896 dB und die Rate 0.041 bpp. 220

Abb. 8.31: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=45. PSNR dieses Bildes ist 30.24 dB und die Rate 0.71 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp. 220

Abb. 8.32: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 31.36 dB und die Rate 0.07 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=44), PSNR diese Bildes ist 30.083 dB und die Rate 0.07 bpp. 221

Abb. 8.33: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 33.13 dB und die Rate 0.68 bpp. Das rechte

Bild wurde komplett mit H.264 codiert (qp=22), PSNR diese Bildes ist 42.955 dB und die Rate 0.656 bpp. 221

Abb. 8.34: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 38.80 dB und die Rate 0.5 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=24), PSNR diese Bildes ist 41.725 dB und die Rate 0.523 bpp. 222

Abb. 8.35: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 42.8 dB und die Rate 1.31 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=16), PSNR diese Bildes ist 47.014 dB und die Rate 1.27 bpp. 222

Abb. 8.36: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 26.38 dB und die Rate 0.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 26.727 dB und die Rate 0.159 bpp. 223

Abb. 8.37: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 28.81 dB und die Rate 0.77 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.073 dB und die Rate 0.746 bpp. 223

Abb. 8.38: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 29.93 dB und die Rate 1 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=26), PSNR diese Bildes ist 38.373 dB und die Rate 0.988 bpp. 224

Abb. 8.39: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 35.86 dB und die Rate 1.64 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=20), PSNR diese Bildes ist 42.934 dB und die Rate 1.66 bpp. 224

Abb. 8.40: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 26.55 dB und die Rate 1.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=27), PSNR diese Bildes ist 37.714 dB und die Rate 1.183 bpp..... 225

Abb. 8.41: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 29.87 dB und die Rate 2.11 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=19), PSNR diese Bildes ist 44.77 dB und die Rate 2.09 bpp. 225

Abb. 8.42: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40.

PSNR dieses Bildes ist 23.44 dB und die Rate 0.24 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 23.338 dB und die Rate 0.22 bpp. 226

Abb. 8.43: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 25.28 dB und die Rate 1.02 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 29.52dB und die Rate 1.009 bpp. 226

Abb. 8.44: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 27.34 dB und die Rate 1.81 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 34.757 dB und die Rate 1.887 bpp. 227

Abb. 8.45: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 31.78 dB und die Rate 2.63 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=24), PSNR diese Bildes ist 38.47 dB und die Rate 2.555 bpp. 227

Abb. 8.46: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=50, PSNR=28.05, bpp=0.73), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=50, PSNR=27.93, bpp=0.74), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=50, PSNR=27.47, bpp=0.75). 228

Abb. 8.47: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=31.24, bpp=0.67), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=30.86, bpp=0.74), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=30.24, bpp=0.71). 229

Abb.8.48: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=33.87, bpp=0.75), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=33.18, bpp=0.75), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=33.13, bpp=0.68). 230

Abb.8.49: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=50, PSNR=24.85, bpp=0.83), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=50, PSNR=23.72, bpp=0.63), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=50, PSNR=23.45, bpp=0.70). 231

Abb. 8.50: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=27.51, bpp=0.83), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=26.67, bpp=0.71), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=26.13, bpp=0.74). 232

Abb. 8.51: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=35, PSNR=32.63, bpp=0.94), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=35, PSNR=31.18, bpp=0.88), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=35, PSNR=31.42, bpp=0.87). 233

Abb.8.52: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Barbara*, Bild *Barbara* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=29.04, bpp=1.09), Bild *Barbara* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=26.96,bpp=1.14), Bild *Barbara* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=26.23, bpp=1.06).. 234

Abb.8.53: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Barbara*, Bild *Barbara* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=35, PSNR=31.88, bpp=1.23), Bild *Barbara* nach Korrektur der Bitebenen bei Methode II (qp=35, PSNR=29.03,bpp=1.30), Bild *Barbara* nach Korrektur der Bitebenen bei Methode III (qp=35, PSNR=28.02, bpp=1.20).. 235

Abb. 8.54: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Baboon*, Bild *Baboon* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=24.34, bpp=1.12), Bild *Baboon* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=23.33,bpp=0.84), Bild *Baboon* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=22.88, bpp=0.74). 236

Abb.8.55: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Baboon*, Bild *Baboon* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=26.34, bpp=1.22), Bild *Baboon* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=25.25,bpp=1.04), Bild *Baboon* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=25.28, bpp=1.02)… 237

Symbolverzeichnis

DSC	verteilte Quellencodierung	Engl: distributed source coding
DIC	verteilte Billdercodierung	Engl: distributed image coding
DCT	diskrete Kosinustransformation	Engl: discrete cosine transform
LDPC	Niedrigdichten-Paritäts-Prüfcode	Engl: low density parity check code
PDF	Wahrscheinlichkeitsdichtefunktion	Engl: probability density function
BPP	Bit pro Pixel	Engl: Bit per Pixel
BSC	das binäre symmetrische Kanal	Engl: binary symmetric channel
ML	die maximale Wahrscheinlichkeit	Engl: Maximum likelihood
SW	Slepian Wolf	
BEC	Binärer Auslöschungskanal	Engl: binary erasure channel
SPA	Der Summen-Produkt-Algorithmus	Engl: Sum Product algorithm
i.i.d	unabhängige identisch Verteilung	Engl: indepedent identical distribution
ASWC	Asymmetrische Slepian-Wolf-Codierung	Engl: Symmetric Slepian-Wolf-Coding
SI	Zusatzinformation	Engl: side information
BER	Bitfehlerrate	Engl: bit error rate
LLR	Logarithmische Wahrscheinlichkeitsverhältnisse	Engl: log-likelihood ratio
c-Knoten	Prüfknoten der Tanner-Graphen	Engl: check nodes
v-Knoten	variable Knoten der Tanner	Engl: variable nodes
APP	aposteriori Wahrscheinlichkeit	Engl: aposteriori probability
IntDCT	die ganzzahlige DCT-Transformation	Engl: integer DCT
MMSE	der Vorhersage des minimalen mittleren quadratischen Fehlers	Engl: minimum mean square error estimator
MSE	der mittlere quadratische Fehler	Engl: mean square error
MSB	das höchstwertige Bit	Engl: the most significant bit
LSB	das niedrigstwertige Bit	Engl: the least significant bit

GF(2)	Galois-Feld mit zwei Elementen	Engl: Galois-Feld with two elements
AWGN	additives weißes gaußsches Rauschen	Engl: Additive white Gaussian noise

1 Einführung

Die verteilte Bildercodierung ist eine neue Bildercodierungsmethode. Sie basiert auf der Slepian-Wolf- und der Wyner-Ziv-Theorie und wurde zu Beginn der 1970er Jahre eingeführt. Bei dem Codierungs-Szenario nach Wyner-Ziv codieren wir zwei korrelierte Quellen X und Y unabhängig voneinander, und zwar mit Hilfe zweier getrennter Encoder, einer für X und einer für Y. Die zwei Encoder senden ihre Bitstreams an den Decoder. Der Decoder empfängt diese zwei Bitstreams. Dabei wird angenommen, dass dem Decoder die Korrelation zwischen den zwei Quellen bekannt ist, in anderen Worten, dass dem Decoder die Wahrscheinlichkeitsdichtefunktion (engl: probability density function =PDF) zwischen den Quellen bekannt ist. In diesem Fall kann der Decoder eine gemeinsame Decodierung unter Nutzung dieser PDF durchführen. Im Kontext der Bildercodierung können die zwei korrelierten Quellen beispielsweise zwei räumlich benachbarte Blöcke eines Bildes sein.

Traditionelle Bildcodierungsschemata nutzen die Intra-Korrelation zwischen den Pixeln nur am Encoder; im Wyner-Ziv-Codierungs-Szenario wird diese Korrelation am Decoder genutzt. Das Wyner-Ziv-Codierungs-Szenario erlaubt dem Encoder eine niedrigere Komplexität und somit einen niedrigeren Energieverbrauch, wohingegen der Decoder komplexer sein muss als beim traditionellen Bildercodierungs-Szenario. Das bedeutet, dass die hohe Komplexität, die mit der Intra-Prädiktion-Aufgabe (ausgeführt am Encoder bei herkömmlicher Intra-Codierung in H.264 und auch bei traditioneller Bildercodierung wie jpeg) verbunden ist, bei der Wyner-Ziv-Codierung an den Decoder verlagert wird, was zu einer niedrigeren Codierungskomplexität auf Kosten höherer Decodierungskomplexität führt.

Die wichtigste Anwendung für die verteilte Codierung liegt in den Drahtlos- und Niedrigenergie-Kommunikationssystemen sowie in Handykameras, weil in diesen Fällen am Encoder eine niedrige Komplexität benötigt wird und oft nur wenig Speicherplatz vorhanden ist.

Die Hauptgegenstände dieser Arbeit sind:

1 - Methoden zur verteilten Bildercodierung für die Transformations-Domäne

2 - Codierung einer Bildhälfte in Wyner-Ziv-Codierung

3 – Durchführung der Wyner-Ziv-Codierung mit niedriger Komplexität durch Nutzung von Transformationen und Kanalcodierung.

1.1 Aufbau der Arbeit

Kapitel 2 behandelt kurz die Grundlagen der Informationstheorie in Bezug auf Entropie, verlustfreie Quellencodierung, die Ratevrzerrung-Funktion, die Kanalcodierungstheorie und die Kanalkapazität.

Kapitel 3 erklärt die Theorie der Wyner-Ziv-Codierung und zeigt symmetrische und asymmetrische Methoden zu ihrer Ausführung. Des Weiteren wird das virtuelle Kanalmodell der verteilten Quellencodierung sowie eine Simulation der Wyner-Ziv- und Slepian-Wolf-Codierung mit Kanalcodes (LDPC und Turbo-Codes) vorgestellt. Abschließend werden die Wyner-Ziv-Codierung für Gaußsche Quellen und für den binären symmetrischen Fall sowie die praktische Quantisierung für die Wyner-Ziv-Codierung erläutert.

In Kapitel 4 wird der LDPC-Code vorgestellt und seine Eigenschaften mit denen des Turbo-Codes verglichen, anschließend werden die Methoden der Codierung und Decodierung der LDPC und die Verringerung der Codierungskomplexität erklärt. Erläutert werden auch der reguläre und irreguläre LDPC, der error floor des LDPC, die density evolution zur Analyse der Grenzwerte des LDPC für einen bestimmten Kanal sowie die Benutzungsweise des LDPC zur Korrektur von Bitebenen.

Kapitel 5 enthält einen Überblick über state-of-the-art Artikel über verteilte Bildcodierung.

Kapitel 6 erklärt unsere verteilten Bildercodierungsmethoden, zeigt deren Leistung im Vergleich zur Intra-Codierung H.264 auf und vergleicht alle drei Methoden bezüglich der subjektiv wahrnehmbaren und objektiven Qualität des Bildes. Der Kompressionsfaktor und die korrigierte Bit-Rate des Wyner-Ziv-Bildes, die Codierungskomplexität und die Decodierungskomplexität werden ebenfalls verglichen.

Kapitel 7 enthält die Zusammenfassung der Arbeit und einen Ausblick auf mögliche Weiterführung.

Kapitel 8 enthält die Anhänge mit den visualisierten Vergleichen zwischen den drei Methoden und der H.264 Intra-Coding.

2 Information zu den theoretischen Hintergründen

2.1 Die Entropie, ein stochastischer Prozess

Informationstheorie ist die Mathematische Theorie der Übertragung und Speicherung von Daten. Die wesentlichen Größen der Informationstheorie (Entropie, gegenseitige Information, relative Entropie) beschreiben das Verhalten von Zufallsvariablen [TT91].

Eine Zufallsvariable ist eine Funktion X, die Ereignisse w reellen Zahlen zuordnet. Die Funktion X ist gegeben durch $\Omega \overset{X}{\Rightarrow}$ R. Hier ist Ω der Umfang der Ereignismenge.

Die Informationstheorie behandelt auch den stochastischer Prozess, der eine Familie von Zufallsvariablen darstellt. Ein stochastischer Prozess beschreibt die Wahrscheinlichkeitsbeziehungen zwischen seinen Zufallsvariablen. Ein stochastischer Prozess ist gegeben durch $X = \{X(t), t \in T\}$.

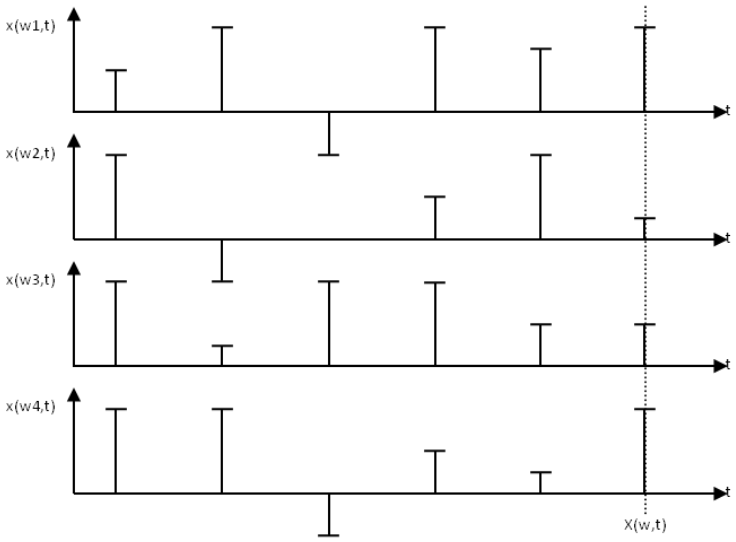


Abb. 2.1: Stochastischer Prozesses besteht aus 4 Zufallsvariablen.

Der stochastische Prozess ist eine Folge von Zufallsvariablen $X(t)$. Das heißt, für jedes t in den Index gesetzt T, $X(t)$ existiert eine Zufallsvariable. In Abb. 2.1 haben wir ein Beispiel für einen stochastischen Prozess bestehend aus 4 Zufallsvariablen $(X(w1,t), X(w2,t), X(w3,t), X(w4,t))$ und Ω=$\{w1,w2,w3,w4\}$. Meist steht t für Zeit und mit $X(w,t)$ wird der Zustand des Prozesses zum Zeitpunkt t bezeichnet.

Wir beschreiben den stochastischen Prozess durch Verteilungsfunktionen und Verteilungsdichtefunktionen.

Wir verwenden den stochastischen Prozess, um Video-Signale und Bildsignale zu beschreiben, weil jeder Pixel des Bildes, also jedes Signal als eine Zufallsvariable in einem stochastischen Prozess behandelt wird und das Video-Signal nichts anderes als eine Familie von stochastischen Prozessen darstellt. Das bedeutet, es gibt Beziehungen zwischen jedem Frame und anderen Frames in dem gleichen Video und es gibt somit Beziehungen zwischen jedem Pixel im Bild und den anderen Pixeln im gleichen Bild.

Die Informationsquelle wird verwendet, um alle Arten von Signale aussendenden Quellen und deren Signalen als Zufallsvariable auszudrücken oder als stochastische Prozesse zu beschreiben.

Die Quelle sendet Symbole, die aus einer alphabetischen Quelle stammen. Wir beschreiben jede Informationsquelle X mit Hilfe ihrer Entropie $H(X)$, die eine Funktion der Wahrscheinlichkeitsfunktion ist. Die Entropie bestimmt die inhärenten Unsicherheiten bei der Verteilungsfunktion einer Zufallsvariable. Die Einheit der Entropie ist Information-Bit pro Symbol. Sie wird beschrieben durch die Gleichung

$$H(X) = \sum_{x \in \mathcal{X}} P(X = x) log_2 \frac{1}{P(X = x)} \qquad 2.1$$

Die gemeinsame Entropie $H(X;Y)$ und die bedingte Entropie $H(X/Y)$ sind einfache Erweiterungen, die die Unsicherheit in der gemeinsamen Verteilung eines Paares von Zufallsvariablen und die Unsicherheit in der bedingten Verteilung eines Paares von Zufallsvariablen wiedergeben. Sie sind definiert durch

$$H(X;Y) = \sum_{\mathcal{X}} \sum_{\Upsilon} p(x,y) log_2 \frac{1}{p(x,y)} \qquad 2.2$$

$$H(X/Y) = \sum_{\mathcal{X}} \sum_{\Upsilon} p(x,y) log_2 \frac{1}{p(x/y)} \qquad 2.3$$

Die Zufallsvariablen X and Y werden jeweils durch die Symbole $\mathcal{X}$ und Υ wiedergegeben.

Die Entropierate des stochastischen Prozess $\{\mathcal{X}_i , i=1:\infty\}$ ist gegeben durch

$$H(\mathcal{X}) = \lim_{n \to \infty} \frac{1}{n} H(X_1, X_2, \dots X_n) \qquad 2.4$$

Hierbei ist n die Anzahl der Zufallsvariablen in dem stochastischen Prozess. Für einen stationären stochastischen Prozess existiert immer eine Entropierate. Die Entropierate ermöglicht einen Vergleich der Entropien von stochastischen Prozessen

unterschiedlicher Länge (bestehend aus unterschiedlichen Anzahlen von Zufallsvariablen).

Die gemeinsame Information $I(X,Y)$ misst die durchschnittliche Realisierung der Zufallsvariablen Y über die Realisierung von X. Das heißt, die Reduktion der Entropie von X kann bestimmt werden, wenn die Realisierung von Y bekannt ist.

$$I(X,Y) = H(X) - H(X/Y) \qquad 2.5$$

Wir bezeichnen den stochastischen Prozess des Bildsignals in vorliegender Dissertation mit X. Das heißt, X ist ein zweidimensionaler stochastischer Prozess. Die Prozesse für Video-Quellen sind dreidimensional.

In dieser Dissertation behandeln wir die digitalen Bilder als zweidimensionale $N \times N$ Matrizen, die die Luminanzwerte des Bildes haben. Jeder Luminanzwert hat 8 Bit, was bedeutet, dass das Bild als Zufallsvariable seinen Wert im Bereich X= (0, 1, 2..., 255) einnimmt. Man kann auch die Bilder als Vektoren in N^2 dimensionalen Raum behandeln, dadurch entspricht jeder Pixel einer separaten Dimension.

2.2 Theorem der fehlerfreien Quellencodierung und die Rateverzerrung -Funktion

In der Informationstheorie beschreibt das Shannon Source Codierungstheorem (oder Theorem der störungsfreien Codierung) die Grenzen der möglichen Datenkomprimierung.

Die Theorie liefert Antworten auf zwei grundlegende Fragen:

1) Was ist die nicht reduzierbare Komprimierung, unterhalb derer ein Signal nicht mehr weiter komprimiert werden kann?
2) Was ist die optimale Übertragungsrate für eine zuverlässige Kommunikation über einen rauschbehafteten Kanal?

Die Antwort über die erste Frage ist das Theorem der Quellencodierung. Der Satz kann wie folgt beschrieben werden:

Gegeben ist eine diskrete gedächtnislose Quelle, deren Entropie $H(X)$ ist. Die durchschnittliche Code-Wort Länge R_X für jede verzerrungsfreie Quellencodierung ist begrenzt, wie in der folgenden Gleichung dargestellt

$$R_X \geq H(X) \qquad 2.6$$

Dieser Satz stellt das mathematische Hilfsmittel für die Bewertung der Datenkomprimierung dar, das heißt für eine verlustfreie Kompression von Daten, die

durch eine diskrete gedächtnislose Quelle generiert werden. Der Quellencodierungs-Satz wird auch als " Theorem der störungsfreien Codierung " bezeichnet, in dem Sinne, dass er die Voraussetzungen, die eine fehlerfreie Codierung ermöglichen, regelt.

Wir können das Theorem der Quellencodierung nicht anwenden für eine kontinuierliche Quelle ohne Codierungsverluste, weil die Beschreibung einer beliebigen reellen Zahl eine unendliche Anzahl von Bits erfordert.

Wir wenden die Rate-Distorsion-Theorie auf die verlustbehaftete Kompression an.

Gemessene Verzerrungen werden durch der Menge der Alphabet-Alphabet Reproduktionspaare aus der Reihe nicht negativer Zahlen dargestellt.

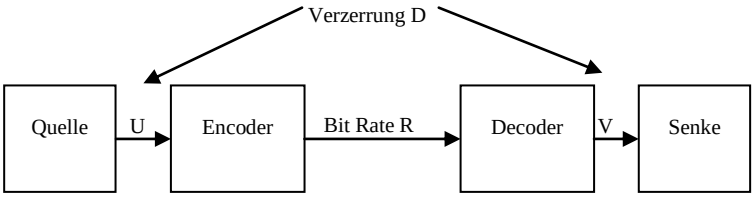


Abb. 2.2: Übertragungssystem.

$$d: x \times \hat{x} \to R^{+} \qquad 2.7$$

In Abb. 2.2 ist das allgemeine Übertragungssystem für verlustbehaftete Kompression dargestellt. x ist in der Gleichung 2.7 das Symbol von Quelle Alphabet U, und $\hat{x}$ ist das Symbol von Quelle Alphabet V. Das heißt, die Verzerrungsfunktion d misst die Kosten für den Austausch von Symbol x durch $\hat{x}$. Die beiden häufigsten Rateverzerrung-Funktion sind:

1) Hamming-Verzerrung, gegeben durch

$$d(x, \hat{x}) = \begin{cases} 0 \text{ wenn } x = \hat{x} \\ 1 \text{ wenn } x \neq \hat{x} \end{cases} \qquad 2.8$$

2) Für den mittleren quadratischen Fehler ist die Verzerrung gegeben durch

$$d(x, \hat{x}) = (x - \hat{x})^2 \qquad 2.9$$

Für das digitale Bild ist die Verzerrung beschrieben durch die Gleichung

$$D_Q = \sum_{i=j=1}^{M\times N} p_{X,Y}(x_i, y_j)(x_i - y_j)^2 \qquad 2.10$$

Hier ist $p_{X,Y}(x, y)$ die Wahrscheinlichkeitsverteilungsfunktion des Bildes, i und j sind Indices der Bildelemente, x und y stellen das Originalbild und das verzerrte Bild dar. Das Bild hat $M \times N$ Elemente.

Die Messung der Verzerrung ist beschränkt, wenn der maximale Wert der Verzerrung d_{max} endlich ist, das heißt im Falle von

$$d_{max} = \max_{x\in X, \hat{x}\in \Upsilon} d(x, \hat{x}) < \infty \qquad 2.11$$

Die Rate-Verzerrungs-Theorie ist der Zweig der Informationstheorie zur Bewältigung des Problems der Bestimmung der minimalen Entropie von Informationen, die über einen Kanal so kommuniziert werden sollen, so dass die Quelle am Empfänger mit einer bestimmten Verzerrung rekonstruiert werden kann.

Die Rateverzerrung-Funktion $R(D)$ für Informationen aus einer Quelle X mit Verzerrung $d(x, \hat{x})$ ist definiert durch

$$R(D) = \min_{p(\hat{x}/x):Ed(x,\hat{x})\leq D} I(X; \hat{X}) \qquad 2.12$$

Wo die Minimierung über alle bedingten Verteilungen $p(\hat{x}/x)$, für die die gemeinsame Verteilung $p(\hat{x}; x)$ die erwartete Verzerrungs-Bedingung $Ed(x, \hat{x}) \leq D$ erfüllt.

Die $R(D)$-Kurve hat zwei wichtige Eigenschaften: Die erste Eigenschaft ist, dass $R(D)$ eine nicht-wachsende Funktion von D ist. Das besagt, dass die Menge der p$(\hat{x}/x)$ nicht bei einer Erhöhung von D zunimmt. Die zweite Eigenschaft ist, dass $R(D)$ konvex ist in D. Die allgemeine Rateverzerrung-Kurve wird dargestellt in Abb. 2.3.

Das Maximum von $R(D)$ ist gleich der Entropie $H(X)$ und tritt auf bei $D = 0$, also bei keiner erlaubten Verzerrung. Für eine gegebene Maxilmalverzerrung D stellt die Rateverzerrung-Funktion $R(D)$ die Untergrenze für die Übertragungsrate in Bit dar.

Wie die obigen Gleichungen 2.12 zeigen, benötigt die Berechnung einer Rateverzerrung-Funktion die stochastische Beschreibung der Quelle X, die durch die PDF-Funktion (engl: probability density function=PDF) dargestellt wird. Dann wird die bedingte PDF gesucht, die die Rate für eine bestimmte Verzerrung minimiert.

Die Berechnung der Rateverzerrung-Funktion erfordert eine detaillierte Kenntnis der Statistik der Quelle (oder des stochastischen Prozesses), welche den Ursprung des betrachteten Signals darstellt. Handelt es sich dabei zum Beispiel um ein Bild, kann die

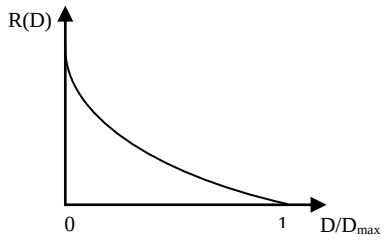


Abb. 2.3: Allgemeine Rate-Verzerrung-Kurve.

Funktion der Rateverzerrung nur bei Kenntnis der Statistik des stochastischen Prozesses, der den Ursprung des Bildes darstellt, errechnet werden.

Der stochastische Prozess erlaubt detaillierte Erkenntnisse über seine Statistik dann, wenn er stationär oder ergodisch ist. In der Praxis sind die meisten stochastischen Prozesse keine stationären oder ergodischen Prozesse. Deswegen können wir ihre Rateverzerrung-Kurven nicht berechnen.

Die Lösung dieses Minimierungsproblems für $R(D)$ ist nur für wenige Fälle möglich, nämlich für die gedächtnislose (unabhängige) Gaußsche Quelle und für die Gaußsche Quelle mit Gedächtnis.

2.2.1 Rateverzerrung-Funktion eines Bildes

Das Bild setzt sich aus $M \times N$ Elementen zusammen, deren Helligkeit jeweils einen von 2^q Werten annehmen kann. Bei dem bekannten Bild kann die Rateverzerrung-Funktion berechnet, daraus wird die Mindestdatenmenge in Bits ermittelt, die zur Wiedergabe des Bildes bei bestimmten Verzerrungseigenschaften nötig ist. Landau und Slepian [LS71] geben vor, jedes Bild als Symbol x aus der Menge X zu betrachten. Die Menge X ist endlich, da es höchstens $2^{q\times M\times N}$ Bilder geben kann. Das bedeutet, dass zur Berechnung der Rateverzerrung-Funktion die Wahrscheinlichkeitsverteilung über $2^{2q\times M\times N}$ Paar-Bilder bekannt sein muss, was die Lösung praktisch unmöglich macht. Aber in der Praxis kommt jedoch eine extreme Änderung der Luminanzwerte so gut wie nie vor.

Das menschliche Auge nimmt ein Bildelement nicht unabhängig vom jeweiligen Kontext wahr. Das Auge ist tolerant gegenüber Verzerrungen von Bildelementen in Bereichen, die reich an kleineren Details sind, aber wesentlich sensibler in Bereichen, die allmähliche Veränderungen von Graustufen beinhalten. Das bedeutet, dass die Rateverzerrung-Kurve für die Bilddaten-Kompression für das menschliche Auge angepasst werden muss. Diese Bildcodierung ist eine wahrnehmungsorientierte Codierung, da das Codierungsschema sich der Wahrnehmung durch das menschliche Auge anpasst.

2.3 Die Kanalkapazität und das Theorem der Kanalcodierung

Das Theorem der Kanalkapazität ist eine der wichtigsten Errungenschaften der Informationstheorie. In einer einzigen Formel stellt es die Wechselwirkungen von drei Schlüsselparametern eines Systems dar:

- Bandbreite des Kanals.
- Die Durchschnittsstärke des übertragenen oder empfangenen Signals.
- Die Stärke des Rauschens am Ausgang des Kanals.

Die Kanalkapazität der Kanal mit Eingang X und Ausgang Y ist als die maximale Menge der Information definiert, die zwischen X und Y ausgetauscht werden kann, wobei das Maximum für alle Möglichkeiten der Eingangsverteilung definiert ist.

Die Kanalkapazität ist beschrieben durch die Gleichung 2.13 , X bezeichnet die Quelle und Y die Senke; mit X ist die Quelle Alphabet bezeichnet.

$$C = \max_{p_X(x)} I(X;Y) \qquad 2.13$$

Man nehme an, dass die Rate eines Fehlerkorrektur-Codes $R = K/n$ beträgt, wobei K die Anzahl von Input-Symbolen und n die Anzahl der Ausgang-Symbole sein soll.

Das Kanal-Codierungstheorem besagt: Angenommen, die Codierungsrate von Übertragung R ist geringer als die Kanalkapazität C und für jede Fehlerwahrscheinlichkeit ϵ ist ein Korrekturcode der Länge n_0 vorgegeben. Es existieren somit Codes der Länge n, die länger sind als n_0, für die die Fehlerwahrscheinlichkeit kleiner ist als ϵ.

Das bedeutet, dass zur Verringerung der Fehlerwahrscheinlichkeit der Encoder mit längeren Vektoren von Signaldaten arbeiten muss (R wird dann wesentlich kleiner als C). Das führt zu größeren Verzögerungen und erfordert höhere Rechenleistungen. Dann, wenn $R \leq C$ kann Übertragung auch bei Störungen fehlerfrei erfolgen.

Die Umkehrung des Theorems ist ebenfalls wahr: wenn $R > C$, können Fehler unabhängig von der verwendeten Codierungstechnik nicht vermieden werden. Die Kanalkapazität wird in Bits pro Kanalbenutzung (Informationsgehalt je Kanalbenutzung) angegeben. Wenn der Kanal nur einmal in einer Sekunde benutzt wird, ist die Einheit somit Bits pro Sekunde.
Durch die Kanalcodierung fügen wir immer Korrektur-Bits zu den ursprünglichen Bits hinzu, um die durch Kanalfehler entstandenen Fehler in der Information korrigieren zu können. Das heißt es steht weniger Energie pro Bit zur Verfügung als bei der Übertragung von Bits ohne Kanalcodierung. Überraschenderweise ist bei Benutzung der Kanalcodierung die Fehlerwahrscheinlichkeit pro Bit geringer als im System ohne Kanalcodierung.

Das Problem der Kanalcodierungstheorie ist, dass sie besagt, dass Codes existieren, die für eine zuverlässige Übertragung genutzt werden können, aber solche praktischen Codes nicht zu finden sind. [Jun55], [Mac03], [JN90] decken diese Theorie sehr gut.

3 Überblick über die verteilte Quellencodierung

Dieses Kapitel hat die Implementierung der Informationstheorie aus dem vorigen Kapitel auf die verteilte Quellencodierung. In den folgenden Abschnitten wird die Theorie der Slepian-Wolf- sowie der Wyner-Ziv-Codierung behandelt und die Benutzung der Kanalcodierung, um die SW- und die WZ-Codierung möglich zu machen. Außerdem werden adaptive Methoden beschrieben, um jeden Punkt innerhalb der SW-Grenze zu erreichen und es werden die praktische WZ-Codierung sowie Simulationsergebnisse vorgestellt um die theoretische Grenze der verteilten Quellencodierung zu bestimmen.

Wir codieren in der Regel die abhängigen Zufallsvariablen unter Nutzung der Ähnlichkeit zwischen ihnen. Dies ermöglicht, dass der Encoder weniger Informationen an den Decoder senden muss, als wenn die Zufallsvariablen unabhängig voneinander codiert würden. Aber es kommen auch Situationen vor, wo der Encoder nicht alle Quellinformationen zur Verfügung hat.

Sehen wir einige dieser Beispiele an:

1. Verschiedene Teile der Quellinformationen können den getrennten Encoder-Terminals zugänglich sein.
2. Decoder können Zugang zu zusätzlichen Informationen über die Quellinformationen haben.

Eine Lösung für Fälle wie diesen bietet die verteilte Quellencodierung (VQC, engl: distributed source coding=DSC). In VQC codieren wir zwei oder mehr abhängige Zufallsvariable auf unabhängige Art und Weise. Das bedeutet, dass wir jede Zufallsvariable bei einem Encoder codieren. Am Ausgang jeder Quelle senden wir einen Bitstrom an den Decoder. Wenn der Decoder die statistischen Abhängigkeiten zwischen den beiden Quellinformationen kennt, führt er die Decodierung aller empfangenen Bitstroms aus.

In Abb. 3.1 wird ein drahtloses Sensorennetzwerk-Szenario dargestellt [PKR02], bei dem der Sensor die gleichen Informationen (es kann die gleiche Szene oder die gleiche Audioinformation sein) aus unterschiedlichen Lagen empfängt. Die Sensoren sind in unterschiedlichen Positionen, aber es existieren Abhängigkeiten zwischen ihren Eingangssignalen, weil die Sensoren ihre Eingangssignale für die gleiche Szene oder die gleiche Audio Quelle empfangen, aber an unterschiedlichen Positionen.

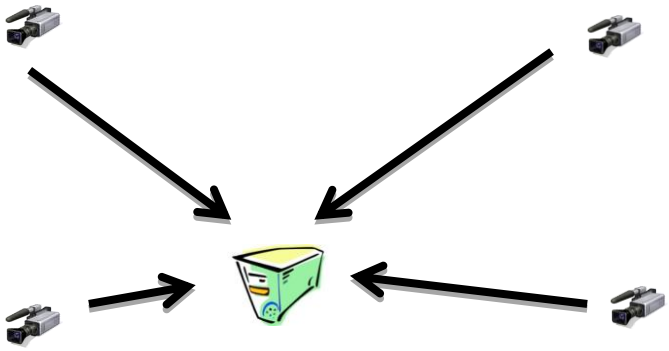


Abb. 3.1: Bildliche Darstellung von verteilter Quellencodierung für mehrere abhängige Sensoren.

Wir sehen, dass es keine Verbindung zwischen den Sensoren gibt. Somit werden wir am Ausgang jedes Encoders einen unabhängigen Bitstrom haben. Der Decoder wird alle diese Bitströme durch die Kenntnis über der Korrelation zwischen all den Eingangssignalen gemeinsam decodieren können.

Wir können die dem Decoder zugänglichen Informationen über die Statistik der Abhängigkeiten zwischen den Eingangssignalen an den Sensoren ausnützen, um den Arbeitsaufwand des Encoders zu verringern. Dies geschieht, indem der Encoder nicht das ganze Eingangssignal sondern nur Teile davon codiert, weil der Decoder die anderen Teile der Signale aus den anderen Bitströmen kennt. In diesem Fall, wenn der Encoder nur einen Teil der Eingangssignale codiert, wird sowohl viel Energie als auch Speicherplatz in den Datenspeichern des Encoders gespart. Diese Einsparung spielt eine große Rolle im drahtlosen Sensorennetzwerk.

Wir können VQC als Folgendes definieren: VQC ist eine Kompressionsmethode, die darauf basiert, dass zwei oder mehr abhängige Sequenzen unabhängig voneinander codiert, aber gemeinsam decodiert werden.

Der Begriff der Verteilung bei VQC bezieht sich nicht auf die Lokalisierung sondern auf den Operationsmodus, ist also nicht als räumliche Verteilung zu verstehen.

3.1 Der virtuelle Kanal in der verteilten Quellencodierung

Wyner betrachtete in [Wyn74] die Beziehung zwischen der verteilten Quellencodierung und der Kanalcodierung.

Bei der verteilten Quellencodierung gehen wir von einer Abhängigkeit zwischen Quelle X und der am Decoder zur Verfügung stehenden zusätzlichen Information Y aus. Im Modell I wird dieser virtuelle Kanal so dargestellt, dass die zusätzliche Information Y als eine durch Kanalfehler verfälschte Version der Quelle X erscheint (sieh Abb. 3.2).

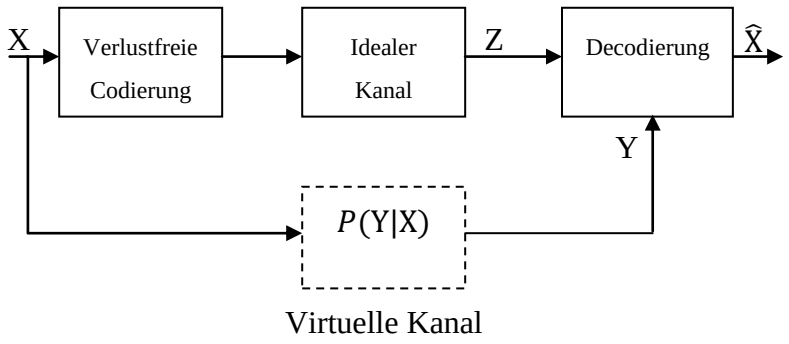


Abb. 3.2: Das virtuelle Kanalmodell I zwischen Quelle X und Zusatzinformation Y.

Im Modell II in Abb. 3.3, was dem in Abb. 3.2 ähnelt, nehmen wir an, dass der Decoder die Bits Z vom Encoder zusätzlich zur Zusatzinformation Y verwendet, um am Decoder eine Rekonstruktion von X zu gewährleisten.

Aus dieser Sicht erscheint es möglich, die Fehler in den empfangenen Sequenzen Y zu korrigieren, indem auf die Sequenzen X ein Kanal-Code angewendet wird. Anders gesagt sieht der Decoder die Bits Z als Paritätsbits (engl:paritybits), die vom Encoder zur Korrektur der Sequenzen von X produziert wurden.

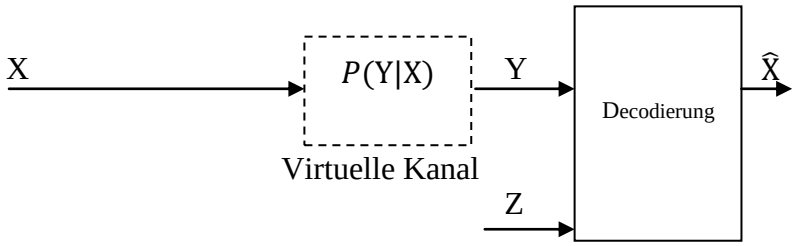


Abb. 3.3: Das virtuelle Kanalmodell II zwischen Quelle X und Zusatzinformation Y.

3.2 Die binäre symmetrische virtuelle Kanal bei der verteilten Quellencodierung

Dieses Modell eines virtuellen Kanals erscheint wirklichkeitsnah für alle Fälle verteilter Quellencodierung. Wir nutzen das Modell des binären symmetrischen Kanals mit der Änderungswahrscheinlichkeit P zwischen der Quelle X und der zusätzlichen Information Y, in dem die Variablen von X und Y unabhängige gleichverteilte Zufallsvariable mit identischer Verteilung sind.

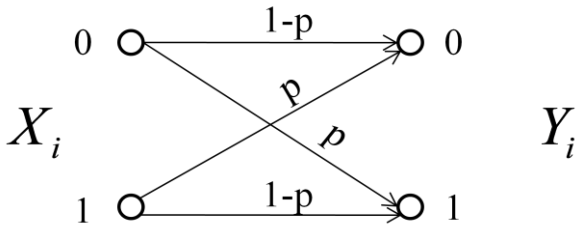


Abb. 3.4: Das binäre symmetrische Kanalmodell (BSC) für unabhängige identisch verteilte binäre Quellen X und Y.

In diesem Modell ist $\{X_i, Y_i\}, i = 1:\infty$ eine Sequenz von korrelierten binären Zufallsvariablen X und Y. X_i und Y_i sind Zufallssequenzen eines endlichen Alphabets von unabhängigen und identisch verteilten (engl:indepedent identical distribution=i.i.d) Samples sind.

In diesem binären symmetrischen Kanal ist $H(X|Y)$ definiert durch

$$H(X|Y) = H(P) = \frac{1}{p} log_2 P + \frac{1}{1-P} log_2 (1-P) \qquad 3.1$$

Die Kanalkapazität ist beschrieben durch

$$C_{BSC} = 1 - H(P) \qquad 3.2$$

Aus der Gleichung 3.1 sehen wir, dass wenn P klein ist, $H(X|Y)$ ebenfalls klein wird, was in diesem Fall bedeutet, dass die Korrelation zwischen den Quellen stark ist.

3.3 Verlustfreie Slepian-Wolf-Quellencodierung

Slepian Wolf-Codierung ist die verlustfreie Komprimierung von Daten aus korrelierten Quellen ohne Kommunikation zwischen den Encodern. In dieser Dissertation wird der Fall von zwei binären korrelierten Quellen X und Y betrachtet, die durch ihre gemeinsame Verteilung gekennzeichnet sind.

Im Falle wenn die beiden Encoder mit einander kommunizieren (wie Abb. 3.5), wird das Minimum verlustfreien Rate für X und Y durch die gemeinsame Entropie $H(X, Y)$ gegeben, wie das Shannon-Theorem [TT91] besagt.

Slepian und Wolf haben bewiesen, dass diese verlustfreie Kompression mit einer verschwindend kleinen beliebigen Fehlerwahrscheinlichkeit für unendlich lange Sequenzen erzielt werden kann. Dies gilt auch wenn die beiden Quellen separat codiert werden, vorausgesetzt, sie werden gemeinsam decodiert, und ihre Korrelation ist

sowohl dem Encoder als auch dem Decoder bekannt. Daher besteht die Herausforderung darin, eine Reihe von nicht miteinander kommunizierenden Encodern, und einen gemeinsamen Decoder, die die theoretische Grenze erreichen.

3.3.1 Asymmetrische Slepian-Wolf-Codierung (ASWC)

Asymmetrische SW-Codierung wird angewendet, wenn eine Quelle, bspw. Y mit der Entropierate H(Y) gesendet wird und als Information zur Decodierung der zweiten Quelle X verwendet wird. In Abb. 3.5 und Abb. 3.6 betrachten wir den Fall, dass X und Y Zufallssequenzen eines endlichen Alphabets von unabhängigen und identisch verteilten (engl:indepedent identical distribution=i.i.d) Samples sind.

Um X und Y separat und verlustfrei codieren und decodieren zu können, werden Raten $R_X \geq H(X)$ und $R_Y \geq H(Y)$ [Sha48] benötigt.

Die SW-Region für zwei diskrete Quellen ist ein Vieleck mit zwei Eckpunkten (Punkte A und B in Abb. 3.7). An diesen Punkten ist eine Quelle (z. B. Quelle Y für Punkt A) bei ihrer Entropierate $R_Y = H(Y)$ komprimiert und kann daher am Decoder rekonstruiert werden unabhängig von der Information, die aus der Quelle X empfangen wurde.

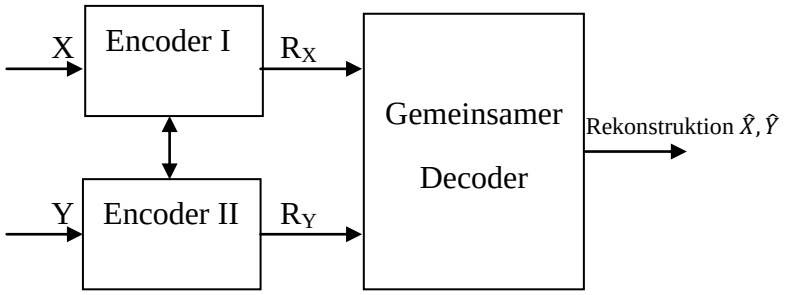


Abb. 3.5: Gemeinsame Codierung und Decodierung von X und Y.

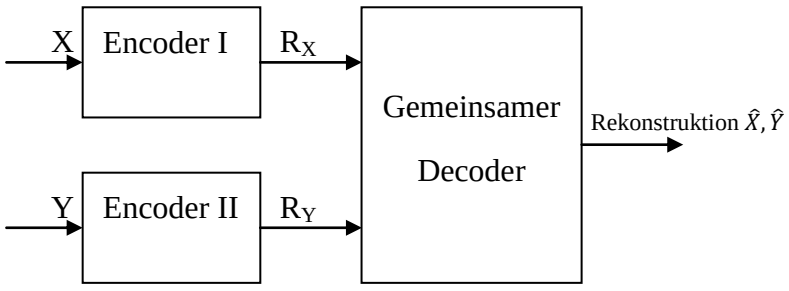


Abb. 3.6: Verteilte Quellencodierung statistisch abhängiger i.i.d. diskreter Zufallssequenzen X und Y.

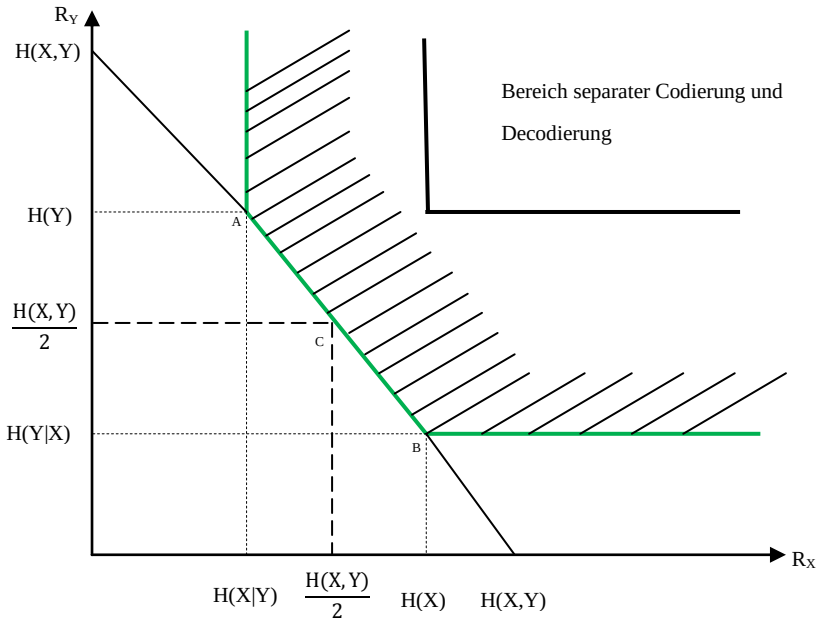


Abb. 3.7: Bereich der erreichbaren Raten von WZ für statistisch i.i.d. diskrete Zufallssequenzen X und Y.

Die Quelle Y wird als Zusatzinformation (engl:side information=SI) bezeichnet, die nur am Decoder verfügbar ist. Die Kompressionsrate von X ist geringer als die Entropie $R_X < H(X)$. Die Kompression ist genauer bei der bedingten Entropie $H(X|Y)$ und die Decodierung kann nur stattfinden, wenn Y dem Decoder zur Verfügung steht.

Die Quellen X und Y spielen verschieden Rollen in diesem Schema, daher wird dieses Schema als asymmetrische SW-Codierung bezeichnet. Der andere Eckpunkt B des SW-Raten-Bereiches in Abb. 3.7 kann annähernd bestimmt werden, indem die Rollen von X und Y vertauscht werden.

Das Slepian-Wolf-Theorem [SW73] zeigt, dass der Ratenbereich (mit beliebig kleiner Fehlerwahrscheinlichkeitstoleranz) erweitert wird bis

$$R_X \geq H(X/Y), R_Y \geq H(Y/X), R_X + R_Y \geq H(X,Y) \qquad 3.3$$

$R_X + R_Y \geq H(X,Y)$ in Gleichung 3.3 zeigt, dass auch in dem Fall wenn die Codierung der korrelierten Quellen unabhängig stattfindet, eine Bitrate $R = R_X + R_Y$, die der gemeinsamen Entropie entspricht, ausreicht. Also muss separate Codierung (theoretisch) nicht zu einem Verlust der Kompressionseffizienz im Vergleich zum herkömmlichen Codierungsverfahren (zentralisierte Codierung) führen.

Es muss angemerkt werden, dass für korrelierte Quellen immer gilt $H(X) + H(Y) > H(X,Y)$, und $R_X + R_Y = H(X,Y)$ genauso gut ist wie die erreichbare Rate für gemeinsame Codierung und gemeinsame Decodierung (d.h. zentralisierte Codierung)

von X und Y. In Abb. 3.7 entspricht die horizontale grüne Linie $R_Y \geq H(Y/X)$, die vertikale grüne Linie $R_X \geq H(X/Y)$ und die diagonale grüne Linie $R_X + R_Y \geq H(X,Y)$. Das bedeutet, dass die grüne Linie die Untergrenze der erreichbaren Rate in der Theorie der verteilten Quellencodierung aufzeigt. Der Bereich, wo $R_X \geq H(X)$ und zugleich $R_Y \geq H(Y)$ ist der Bereich für zentralisierte Codierung.

Definition: A $(2^{nR_1}, 2^{nR_2}, n)$ verteilter Quellencode für die gemeinsame Quelle (X,Y), bestehend aus zwei Encoderfunktionen,

$$f_1: \mathrm{X}^n \rightarrow \{1,2,\ldots,2^{nR_1}\}$$

$$f_2: \mathrm{Y}^n \rightarrow \{1,2,\ldots,2^{nR_2}\}$$

und einer Decoderfunktion

$d: \{1,2,\ldots,2^{nR_1}\} \times \{1,2,\ldots,2^{nR_2}\} \rightarrow \mathrm{X}^n \times \mathrm{Y}^n$

Wo (R_1, R_2) als Ratenpaar des Codes bezeichnet wird

Definition: Die Fehlerwahrscheinlichkeit für eine verteilte Quellencodierung ist definiert als

$$P_e^n = P\left(d\left(f_1(X^n), f_2(Y^n)\right) \neq (X^n, Y^n)\right)$$

Definition: Ein Ratenpaar (R_1, R_2) wird als erreichbar für die zwei Quellenterminale $\{(X_i, Y_i)\}, i = 1:\infty$ angesehenen, wenn eine Sequenz von $(2^{nR_1}, 2^{nR_2}, n)$ verteilten Quellencodes mit $P_e^n \rightarrow 0$ existiert.

Beispiel

Im Beispiel gehen wir aus von diskreten, gedächtnislosen, gleich verteilten Quellen X und Y, die die Variablenmenge *x* und *y* von 7 Bits Länge erzeugen. Die Hamming-Distanz zwischen *x* und *y* beträgt maximal 1.

Wenn X und Y jeweils unabhängig codiert und unabhängig decodiert werden, haben wir (sieh Gleichung 2.1)

$$\mathrm{R_x} = \mathrm{H(X)} = \mathrm{R_y} = \mathrm{H(Y)} = 128\left[\frac{1}{128} \times \log_2(128)\right] = 7 \text{ bits}$$

In diesem Fall müssen wir senden

$$\mathrm{R} = \mathrm{R_x} + \mathrm{R_y} = 14 \text{ bits}$$

Wenn wir X und Y gemeinsam codieren und decodieren, müssen wir senden (sieh Gleichung 2.2)

$$\mathrm{R} = \mathrm{H(X,Y)} = 10 \text{ bits}$$

In Abhängigkeit vom SW-Theorem können wir für diese Quellen die verteilte Quellencodierung anwenden, wenn (sieh Gleichung 3.3).

1) $R_x \geq H(X/Y) = 3$ bits (sieh Gleichung 2.3).
2) $R_Y \geq H(Y/X) = 3$ bits (sieh Gleichung 2.3).
3) $R_x + R_Y \geq H(X,Y) = 10$ bits.

Das bedeutet, die theoretische Grenze liegt bei 10 Bits und bei der symmetrischen VQC (Punkt c in Abb. 3.7) werden 5 Bits für jede Quelle gesendet.

Andere Raten-Paare mit $R_x + R_Y = 10$ Bits stellen asymmetrische Situationen dar mit unterschiedlichen Bitraten-Verteilungen zwischen X und Y, wo $R_x = H(X) = 7$ bits, $R_Y = H(Y/X) = 3$ Bits (Punkt B in Abb. 3.7) und $R_Y = H(Y) = 7$ bits, $R_X = H(X/Y) = 3$ bits (Punkt A in Abb. 3.7) zwei Extremfälle darstellen mit am Decoder verfügbarer Zusatzinformation (engl: Sideinformazion= SI).

Slepian-Wolf-Codierung wird in der Literatur erwähnt als verlustfreie verteilte Quellencodierung, da sie davon ausgeht, dass zwei statistisch abhängige Sequenzen, die unabhängig codiert wurden, mit einem beliebig kleinen Fehler an einem gemeinsamen Decoder rekonstruiert werden.

Das Problem der Slepian-Wolf-Codierung ist, dass eine generelle theoretische Lösung bisher nicht zu existieren scheint, in dem Sinne dass diese Lösung für eine beliebige Anzahl Quellen von nichtbinären alphabetischen Quellen möglicherweise mit Gedächtnis anwendbar wäre.

Die meisten Entwürfe gehen nicht von Quellen mit Gedächtnis und/oder Gedächtnis in der Korrelation zwischen den Quellen aus.

Die Autoren in [GFZ03] berücksichtigen eine Gedächtniskorrelation zwischen den Quellen. In [Cov75] wird die Anwendbarkeit der Slepian-Wolf-Codierung generell angenommen für beliebige ergodische Prozesse und eine beliebige Anzahl korrelierter Quellen. In [PR00a] und [GF01], [PR00b] werden direkte Methoden beschrieben, den Mittelpunkt c in Abb. 3.7 zu erreichen. In [SPR03] und [SLXG04], [CLME04], [PR99] gibt es Code-Designs, mit denen jeder Punkt zwischen den Punkten A und B in Abb. 3.7 bestimmt werden kann. In [AK75] und [KM79] ist nur die Wiederherstellung von X und X + Y modulo 2 wichtig. In [Wyn75] nutzte der Autor einen anderen separaten Encoder, er nutzte in seinem Paper 3 separate Encoder, die Information eines davon ist die zusätzliche Information im Decoder. In [Tun07], [YB06], [Sim02], [KTRR03], beschäftigten sich die Autoren mit dem Fall, dass die Fehlerwahrscheinlichkeit genau bei Null liegen sollte.

3.3.1.1 Slepian-Wolf-Codierung für multiple Quellen

Die Methode der Slepian-Wolf-Codierung kann auf multiple Quellen ausgeweitet werden.

In [LLN+04] entwarf der Autor eine SWC für mehr als zwei Quellen. In [SLXG04] und [SLXG06] weiteten die Autoren einen Code-Entwurf auf M Quellen aus.

Für eine verlustfreie Codierung von n Quellen (sieh Abb. 3.8) sind die Eckpunkte des Bereiches erreichbarer Raten (im Fall von zwei Quellen waren dies A und B in Abb. 3.7) $[H(X_1), H(X_2|X_1), H(X_3|X_1,X_2), \ldots, H(X_n|X_1,X_2,\ldots,X_{n-1})]$.

Aus der Kettenregel für die gemeinsame Entropie geht hervor, dass

$$\sum_{i=1}^{n} H(X_i|X_1,\ldots,X_{i-1}) = H(X_1,\ldots,X_n) \qquad 3.4$$

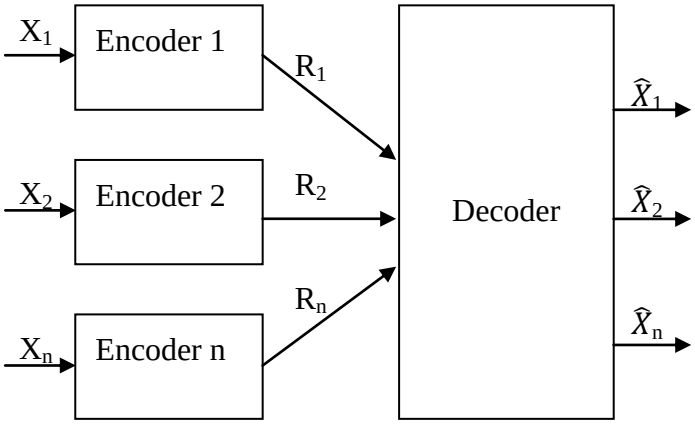


Abb. 3.8: Verteilte Quellencodierung für mehrere Quellen.

Bei der Annahme dass X_1 am Decoder als zusätzliche Information verfügbar ist, aus der Ausweitung des SW-Codierungstheorems [SW73] auf multiple Quellen [Wol73], [Cov75], müssen die Codierungsraten $R_2,R_3,\ldots,R_n$ all diese Gleichungen erfüllen.

$$R_2 \geq H(X_2|X_1,X_3,X_4,X_5 \ldots,X_n) \qquad 3.5$$

$$R_3 \geq H(X_3|X_1,X_2,X_4,X_5 \ldots,X_n) \qquad 3.6$$

$$R_n \geq H(X_n|X_1,X_2,\ldots,X_{n-1}) \qquad 3.7$$

$$R_2 + R_2 + \cdots + R_n \geq H(X_2,X_3,X_4 \ldots,X_n|X_1) \qquad 3.8$$

Wenn wir versuchen, eine der Raten R_2, R_3, …, R_n, zu minimieren, was bedeutet, dass wir eine der Raten (z.B. R_2) an die theoretische Grenze $H(X_2|X_1,X_3,X_4 \ldots,X_n)$ bringen, geht aus den Gleichungen 3.5 bis 3.8 hervor, dass die Raten R_3,$R_{4,\ldots,}R_n$ die folgende Ungleichung erfüllen müssen

$$R_3 \geq H(X_3|X_1, X_4, X_5 \dots, X_n) \quad 3.9$$

$$R_4 \geq H(X_4|X_1, X_3, X_5 \dots, X_n) \quad 3.10$$

$$R_n \geq H(X_n|X_1, X_3, X_4 \dots, X_{n-1}) \quad 3.11$$

Und die Gleichung 3.8 muss lauten

$$R_{i_1} + \cdots + R_{i_k} \geq H\left(X_{i_1}, \dots, X_{i_k} \middle| X_{j_1}, \dots, X_{j_{l-k}}\right) \quad 3.12$$

Mit $k \leq l,\ \{i_1, \dots, i_k\} \subseteq \{1, \dots, l\}$ und $\{j_1, \dots, j_{l-k}\} = \{1, \dots, l\} | \{i_1, \dots, i_k\}$. Dies korrespondiert mit dem Eckpunkt des erreichbaren Bereiches.

Das binäre symmetrische Modell, welches auf den virtuellen Kanal zwischen der Quelle und der zusätzlichen Information angewendet wird, kann außerdem auf mehr als zwei Quellen ausgeweitet auch auf nicht-binäre Quellen angewandt werden. [LLN+04].

Die symmetrische Slepian-Wolf-Codierung haben wir insofern besprochen, als dass alle Methoden sich auf die Bestimmung der Eckpunkte bezogen.

Das Problem der symmetrischen Codierung liegt darin, dass sie nur auf die statistischen Quellenmodelle angewendet wurde mit binären gleichwahrscheinlichen Ausgänge, deren Korrelation von ihrem virtuellen BSC Kanal (engl:binary symmetric channel) komplett erfasst wurde. Dieses Modell ist nicht anwendbar auf mehr als zwei Quellen, manchmal auch nicht auf zwei. Zum Beispiel nehmen wir an in Abb. 3.8, dass eine der Quellen (z.B. X_2) als zusätzliche Information am Decoder zur Verfügung steht und wir nehmen an, dass das Modell der Korrelation zwischen den Quellen ein binäres symmetrisches Modell mit n Quellen ist. Der Versuch, die symmetrische SW-Codierung auf andere Quellen X_1, X_3, X_4, …, X_n anzuwenden und einen beliebigen Punkt an der theoretischen Grenze zu bestimmen – vorgegeben durch die Gleichungen 3.9, 3.10, 3.11 – wirft in diesem Fall das Problem auf, dass es nicht klar ist, wie die symmetrische SW-Codierungsmethode für zwei Quellen modifiziert werden kann, um für mehr als zwei Quellen anwendbar zu sein.

Die symmetrische SW-Codierung kann nicht als genauso effizient betrachtet werden, wie die asymmetrische, da die symmetrische SWC nicht effizient angewendet werden kann, die asymmetrische dafür jede Art Korrelation zwischen den Quellen und der Zusatzinformation abdeckt.

3.3.1.2 ASWC mit der Syndrom-Methode

Der Beweis für das SW-Theorem basiert auf dem Konzept der Code-Aufteilung (engl:code binning). Dies bedeutet die Aufteilung der Input-Räume aller möglichen

Quellen-Outputs in disjunkte Gruppen, die wir als Teilmengen (engl:bins) bezeichnen und deren Vektoren möglichst weit auseinander liegen sollen (das heißt, die Codes müssen eine möglichst große Hamming-Distanz haben).

In [Wyn74] schlug Wyner vor, diese Teilmengen als Nebenklassen (engl:cosets) eines binären linearen Codes für das spezifische Korrelationsmodell zu konstruieren und zeigte die Optimalität dieser Konstruktion.

Die Optimalität der Konstruktion bedeutet, dass wenn der lineare Block-Code die Kapazität des binären symmetrischen Kanals erreicht, der die Korrelation der Quellen darstellt, kann der Kapazität ausfüllende Code in einen SW-ausführenden Code umgewandelt werden. In anderen Worten, wenn die Rate dieses linearen Block-Codes der Quelle X sich der Kapazität des Korrelationskanals annähert, hat der lineare Block-Code eine Rate nahe den SW-Grenzen. Wir haben in diesem Fall den besten theoretischen Bereich der SW-Kurve in Abb. 3.7, was bedeutet, dass wir in Abb. 3.7 die horizontale und die vertikale grüne Linie erreichen.

Für einen guten linearen Block-Code gilt, dass je größer die Hamming-Distanz der Codewörter ist, desto besser wird der Code. Der Code wird verbessert, wenn alle Codewörter die maximale Hamming-Distanz aufweisen.

In [JC83] findet man Informationen über den binären Hamming-Kanalcode.

Die Komplexität des Block-Codes (n, k) im Hinblick auf Speicherung und Decodierung seiner Signale kann reduziert werden, wenn der Code linear ist. Für den linearen Code gilt, dass die Summe zweier Codewörter ebenfalls ein Codewort darstellt. In diesem Fall erfordert die Wiedergabe des Codes nur die Speicherung von k Vektoren der Länge n, während für die Speicherung aller Codewörter eines nicht-linearen Codes q^k Vektoren notwendig wären, Wobei q die Anzahl der Quellensymbole des Codes ist.

Wir sprechen zuerst über die Eigenschaften von linearen Block-Codes und darüber, wie wir diese einsetzen können, um am Decoder fehlerhafte Code-Wörter zu zu korrigieren. Danach entwickeln wir den Gedanke weiter bei zur Ausführung der asymmetrischen SWC.

Wir definieren den linearen Block-Code (n, k) durch seine Paritäts-Prüfmatrix (engl:parity check matrix) H, wobei die Dimension von H $(n - k) \times$ n ist oder über seine Generatormatrix G mit der Dimension $k \times n$. Die Beziehung zwischen H und G wird durch die Gleichung wiedergegeben

$$G \times H^T = 0 \qquad 3.13$$

Wobei $\times$ für Multiplikation steht bei einem Galois-Feld mit zwei Elementen (Abkürzung:GF(2)) für den binären linearen Code. Die Paritäts-Prüfmatrix H für den

Code enthält die Information über die minimale Distanz für den Code. Die minimale Distanz für den linearen Code (n, k) wird eingegrenzt durch

$$d_{min} \leq n - k + 1 \tag{3.14}$$

Ein Code, dessen Korrekturvermögen im Hinblick auf zufällige Fehler mit *t* angegeben wird, braucht eine Mindestdistanz d_{min} zwischen den Codewörtern, die die folgende Gleichung erfüllt:

$$d_{min} \geq 2t + 1 \tag{3.15}$$

Das Codewort ergibt sich aus der Gleichung

$$C = m \times G \tag{3.16}$$

Hier ist m die Nachricht der Länge von k Bits, die codiert werden müssen. Der Empfänger hat das korrekte Codewort, wenn es die folgende Gleichung erfüllt

$$C \times H^T = 0 \tag{3.17}$$

Der Code des linearen Block-Codes wird definiert durch

$$\delta = \{m: m \times H^T = 0\} \tag{3.18}$$

Die Code-Rate des linearen Block-Codes ist gegeben durch

$$R = k/n \tag{3.19}$$

Der erste Gedanke ist, dass die Implementierung des linearen Block-Codes bei der ASWC Folgendes voraussetzt:

Wir gehen vom folgenden Modell aus: Es existieren zwei binäre symmetrische Quellen, Input *X* und Output *Y*, wobei der BSC mit der Fehlerwahrscheinlichkeit *p* behaftet ist. Wir nehmen an, *X* sei ein linearer Code, jede von dessen Sequenzen *n* Bits lang ist. Dies bedeutet, dass wir annehmen, *Y* sei der verfälscht empfangene lineare Code von *X*. Wenn der Decoder *Y* empfängt, wäre nur in dem Fall sicher feststellbar, welches Code-Wort der Input war, wenn die Anzahl der Code-Wörter von *X* $2^{n(1-H_2(p))}$ nicht übersteigt. Wenn dies doch der Fall sein sollte, würde die Hamming-Distanz zwischen den Code-Wörtern nicht größer/gleich $2(d_{maxH}(X, Y)) + 1$ (t in der Gleichung 3.15 wird $t = d_{maxH}(X, Y)$) sein, was dazu führen würde, dass der Empfänger *X* nicht aus *Y* ableiten könnte.

Wir führen das beschriebene Verfahren durch, um *X* zu decodieren, nachdem das Code-Wort *Y* empfangen wurde und greifen dabei auf die Syndrom-Codierung zurück.

Wir entnehmen das Syndrom aus der Gleichung 3.20, wo u eine Sequenz der Länge *n* darstellt. Wenn u eine Sequenz aus der Quelle *X* ist (in diesem Fall befindet sich u im Teilmenge δ), wird das Syndrom S gleich Null sein.

$$S = u.H^T \tag{3.20}$$

Die Sequenzen, die das gleiche Syndrom aufweisen, also die Nebenklasse (engl: coset), ist definiert durch

$$\beta = \{u: u \times H^T = S\} \tag{3.21}$$

Die Code-Gruppe δ in der Gleichung 3.18 ist eine der Teilmengen der Nebenklasse β in der Gleichung 3.21, aber δ und β sind Teilmengen in Raum $\mathcal{F}_q^n$. Wobei q gleich 2 ist bei Nutzung binärer Sequenzen.

Wenn ein Code-Wort m aus der Teilmenge δ über einen binären Kanal mit Änderungswahrscheinlichkeit p versandt wird, ist die empfangene Sequenz fehlerhaft und gegeben durch

$$y = m + e \tag{3.22}$$

Wo e die Fehlersequenz ist. Da der Kanal binär ist, ist der Operator + in der Gleichung in GF (2) gegeben.

Das Syndrom des empfangenen Vektors y ist gegeben durch

$$\hat{S}_y = y \times H^T \tag{3.23}$$

Und da der Code linear ist, haben wir

$$\hat{S}_y = (m \times H^T) + (e \times H^T) = 0 + (e \times H^T) \tag{3.24}$$

Daher ist das Syndrom eines Code-Wortes aus der Teilmenge δ stets Null (sieh Gleichung 3.18).

Alle Code-Wörter mit dem gleichen Fehlermuster e haben das gleiche empfangene Syndrom $\hat{S}_y$. Das bedeutet, das Syndrom ist hilfreich, um herauszufinden ob die empfangene Sequenz y ein Code-Wort ist oder durch Störungen verfälscht wurde. Daher brauchen wir nur die Syndrome und die dazugehörigen Fehlermuster zu speichern. Die gespeicherte Tabelle wird die Syndrom-Decodierungs-Tabelle genannt.

Die auf *maximum likelihood* basierende Decodierung, findet statt, indem nach dem Code-Wort $m\epsilon\delta$ gesucht wird, welches die geringstmögliche Hamming-Distanz zu y hat. Dabei wird die folgende Gleichung benutzt

$$\hat{m} = \arg\min_{m\epsilon\delta} d_H(m, y) \qquad 3.25$$

Das heißt, dass beim Empfangen eines Code-Worts y der Empfänger zunächst dessen Syndrom berechnen wird und danach nach einem Code-Wort mit dem gleichen Syndrom sowie der minimalen Hamming-Distanz zu y sucht.

Wir erhalten genau die Gleichung 3.25, wenn wir für jedes Syndrom einen Nebenklassenführer (engl: coset leader) berechnen, bei dem es sich um einen Index des Syndroms handelt. Ein Syndrom-Decoder berechnet zuerst das Syndrom der empfangenen Sequenz aus $S = y \times H^T$(Dieses Syndrom wird nur für eine Nebenklasse Null sein, für andere nicht), dann berechnen wir den Nebenklassenführer durch die Funktion $l(S)$. Dieser Nebenklassenführer stellt den Schätzwert der maximalen Wahrscheinlichkeit (engl:Maximum likelihood =ML) des Fehlermusters dar. Der Maximum-Likelihood-Schätzwert von m ist gegeben durch

$$\hat{m} = y \oplus l(y.H^T) \qquad 3.26$$

Das Problem der Syndrom-Decodierung ist, dass sie nicht unmittelbar auf die verteilte Quellencodierung angewendet werden kann, da die Sequenz im Kanal-Input (Sequenz der Quelle *X*) nicht zwangsläufig ein Code-Wort (eins der $2^{n(1-H_2(p))}$ Code-Wörter) des Teilmenge δ ist. Da die Sequenz von *X* jeder der 2^n binären *n-Vektoren* sein kann, muss die Syndrom-Herangehensweise weiter entwickelt werden.

Wyner schlug in [Wyn74] vor, Teilmengen (engl:bins) als Nebenklassen (engl:cosets) eines Kapazitäts-erfüllenden Paritäts-Prüfcodes zu konstruieren. Laut diesem Vorschlag unterteilt der Code den Raum mit 2^n Input-Wörtern in 2^{n-k} Nebenklassen. Jede Nebenklasse hat 2^k Input-Wörter der Länge n. Jede Teilmenge stellt einen Nebenklassen-Code (engl:coset code) des linearen binären Block-Codes dar, was bedeutet dass die Hamming-Distanzen des Originalcodes in jeder Teilmenge (engl:bin) erhalten bleiben.

Die Distanz zwischen jeweils zwei Nebenklassen entspricht der minimalen Hamming-Distanz. Weil wir 2^{n-k} Nebenklassen haben, können wir sie mit dem distinkten Syndrom der Länge $n-k$ bezeichnen. Für das Modell des BSC zwischen *X* und *Y* bedeutet dies, dass wir in jeder Nebenklasse $2^{n(1-H_2(p))}$ Code-Wörter haben, es heißt auch, dass $2^{nH_2(p)} = 2^n/2^{n(1-H_2(p))}$ Nebenklassen in diesem Fall vorhanden sind.

In Abb. 3.9 wird die mit der Sydrom-Methode dargestellte Systemstruktur der ASWC im Blockdiagramm gezeigt. Es wird von zwei Quellen *X* und *Y* ausgegangen, die die

Sequenzen *x* und *y* senden, welche als korrelierte binäre Sequenzen der Länge n gedacht werden. Der Encoder berechnet das Syndrom von $x: S = x.H^T$ und sendet es an den Decoder, das bedeutet, dass der Encoder die Sequenz *x* von *n* Bits Länge auf ihr Syndrom der Länge $n-k$ Bits reduziert, was zu einer Kompressionsrate von $n/n-k$ führt.

Hier gehören nicht alle *x* zur gleichen Code-Gruppe δ der Nachricht m in Gleichung 3.18, was bedeutet, dass das Syndrom S in diesem Fall nicht immer Null ist.

Der Decoder sucht nach der Sequenz in der Nebenklasse, die sich am nächsten an *y* befindet bei der gegebenen Korrelation zwischen den Quellen *X* und *Y* und dem empfangenen Nebenklassen-Index *S*. Die Gleichung für die Ausführung dieser ML-Decodierung ist

$$\hat{x} = \arg\min_{x \epsilon C_s} d_H(x, y) \qquad 3.27$$

In 3.27 sind C_s die Code-Wörter von *X*, die das gleiche Syndrom aufweisen (das Syndrom ist nicht notwendigerweise Null, was bedeutet, dass C_s nicht notwendigerweise δ in Gleichung 3.25 entspricht).

Die ML wird hier an einer Menge Vektoren durchgeführt, die sich innerhalb der Teilmenge dieses Syndroms befindet.

Wenn der Decoder das Fehlermuster entdeckt, welches dem Nebenklassenführer entspricht, fügt er es der Zusatzinformation y hinzu durch die Gleichung

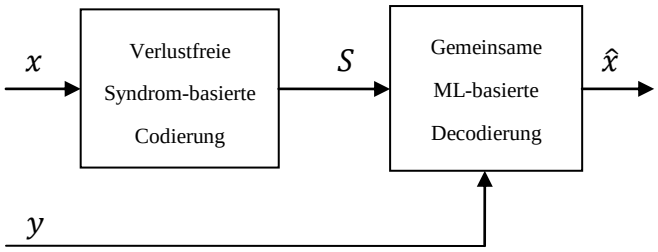


Abb. 3.9: ASWC mit der Syndrom-Methode.

$$\hat{x} = y \oplus l(x.H^T) \qquad 3.28$$

Man bemerke, dass im Falle eines binären linearen Block-Codes der Operator $\oplus$ als Summenoperator in modulo 2 verwendet wird.

Der Schlüssel zu einer optimalen Decodierung liegt im Design der Nebenklasse. Um die besten Nebenklassen für die verteilte Codierung in unserem Fall zu finden, brauchen wir eine Mindest-Hamming-Distanz zwischen den Code-Wörtern in jeder Teilmenge

(also eine Mindest-Hammingdistanz für den linearen Code), die das Doppelte der größten Hamming-Distanz zwischen der Sequenz *x* aus der Quelle *X* und deren korrelierter Sequenz *y* aus der Quelle *Y* beträgt.

$$d_{minH}(\delta) \geq 2d_{maxH}(x,y) + 1 \qquad 3.29$$

Das erste praktische Syndrom-Schema heißt DISCUS [PR03]. In [LXG02] nutzt der Niedrigdichten-Paritäts-Prüfcode-Decoder (engl: low density parity check code decoder =LDPC) das Syndrom. Die Autoren in [RLG07] nutzen den Faltungscode (engl:Convolutions-Code) und den Turbo-Code für die SW-Codierung. In [TLB05] nutzt der Autor den Viterbi-Decoder.

Die meisten Designs berücksichtigen keine Quellen mit Gedächtnis und/oder Gedächtnis in der Korrelation zwischen den Quellen. Die einzige Methode, die Korrelation mit Gedächtnis berücksichtigt, ist [GFZ03].

Beispiel der ASWC mit der Syndrom-Methode

Gehen wir von zwei binären Quellen *X* und *Y* aus, wobei die Sequenzen beider Quellen die Länge von 3 Bits haben und die Wahrscheinlichkeitsdichtefunktion gleich verteilt sein sollen. Das bedeutet $H(X) = H(Y) = 3$ Bits. Zur Ausführung von verteilter Codierung und verteilter Decodierung müssen $H(X) + H(Y) = 3 + 3 = 6$ Bits an den Decoder gesandt werden.

Angenommen die Korrelation der beiden Quellen ist gegeben durch $d_{maxH}(x,y) \leq 1$. Das bedeutet, wenn $y = 100$, kann *x* 100, 101, 110, 000 sein. Das hat zur Folge, dass bei Nutzung eines gemeinsamen Encoders und eines gemeinsamen Decoders der Encoder nur $R = 2$ Bits sendet.

Die Frage ist: wie können wir mit $R = 2$ Bits auskommen bei verteilter Codierung und gemeinsamem Decodier, wobei die Quelle *Y* am Decoder verfügbar ist?

Die Antwort ist:

Für den asymmetrischen Fall bei der verteilten Quellencodierung stellen wir die Sequenzen der Quelle *X* als binäre Wörter eines binären Block-Codes dar. In diesem Fall ist $n = 3$ Bits. Zum Aufbau eines guten linearen Block-Codes für die Syndrom-Decodierung sollte sich die minimale Hamming-Distanz des Block-Codes aus der Gleichung 3.29 ergeben

$$d_{minH}(\delta) = d_{minH}(x_i, x_j) \geq 2d_{maxH}(x,y) + 1 \Longrightarrow d_{minH}(\delta) \geq 3$$

Wobei x_i und x_j zwei beliebige Sequenzen der Quelle X innerhalb einer Teilmenge sind. Wir nehmen auch an, dass die Quelle Y nur am Decoder verfügbar ist.

Wir teilen die Code-Wörter von X in disjunkte Nebenklassen auf, die minimale Hamming-Distanz zwischen den Code-Wörtern in jeder Nebenklasse sollte 3 sein. Beispielsweise kann es sich um folgende Nebenklassen handeln:

{000,111},{001,110},{010,101},{100.011}.

Die Nebenklasse {000,111} ist die Teilmenge δ in der Gleichung 3.18, und alle Nebenklassen ergeben die große Teilmenge β in der Gleichung 3.21.

Um zu codieren, welcher Nebenklassen-Index zur Sequenz *x* gehört, braucht der Encoder nur 2 Bits zu senden. Abhängig von der Syndrom-Decodierung sollte der lineare Block-Code $n-k=2 \Longrightarrow 3-k=2 \Longrightarrow k=1$ aufweisen. Das bedeutet, der Encoder sollte die lineare Block-Codierung mit $(n,k,d_{min})=(3,1,3)$ nutzen. Die folgende Paritäts-Prüfmatrix arbeitet mit diesen Merkmalen des linearen Block-Codes.

$$H=\begin{bmatrix}1 & 1 & 0\\ 1 & 0 & 1\end{bmatrix}$$

Und die Generator-Matrix G ist gegeben durch die Gleichung 3.13

$$G=[1 \quad 1 \quad 1]$$

Der Encoder berechnet das Syndrom der Sequenz *x* aus $S=x.H^T$. In unserem Beispiel sind das Syndrom und das Fehlermuster für jede Nebenklasse (engl:coset):

- Nebenklasse :{000,111}$\Longrightarrow$ syndrom $S=00$, Fehlermuster $e=000$
- Nebenklasse :{001,110}$\Longrightarrow$ syndrom $S=01$, Fehlermuster $e=001$
- Nebenklasse :{010,101}$\Longrightarrow$ syndrom $S=10$, Fehlermuster $e=010$
- Nebenklasse :{100.011}$\Longrightarrow$ syndrom $S=11$, Fehlermuster $e=011$

Dann überträgt der Encoder den Nebenklassen-Index, bei dem es sich um das Syndrom handelt. Wir nahmen an, dass der Decoder die Korrelation zwischen den Quellen *X* und *Y* kennt und stellten die Korrelation dar als $d_{maxH}(x,y)\leq 1$.

Der Decoder empfängt das Syndrom vom Encoder und sucht unter allen Code-Wörtern des Syndroms nach demjenigen, welches die geringste Hamming-Distanz zur verfügbaren Sequenz y der Quelle *Y* aufweist. Das bedeutet, dass die Decodierung, die auf der Gleichung 3.27 beruht, $\hat{x}$ finden wird. Die zu sendende Sequenz *x* kann z. B. 110 sein, und zur Nebenklasse {001,110} gehören, der Encoder wird in diesem Fall das Syndrome $S=01$ senden. Wenn die Sequenz von y dann 111 ist, und der Decoder weiß, dass $d_{maxH}(x,y)\leq 1$ ist, was bedeutet, dass die korrekte Sequenz *x* in dem Fall 110 sein muss (Der Decoder kann dies nach Gleichung 3.27 durchführen, welche zu 3.28 äquivalent ist).

Theorem der SWC mit Syndrom für zwei Quellen

Jedes Paar von korrelierten, gleich verteilten Quellen *X* und *Y* mit binären Sequenzen der Länge n und der maximalen Hamming-Distanz t zwischen den den Sequenzen x und y, kann separat komprimiert werden, so dass

1) $R_X \geq n - k$
2) $R_Y \geq n - k$
3) $R_X + R_Y \geq 2n - k$
4) $k \leq n - log_2 \left[\sum_{i=0}^{t} \binom{n}{i}\right]$

Wir können dies unter Nutzung eines linearen binären Codes $(n, k, 2t + 1)$ durchführen. Im Beispiel der ASWC mit der Syndrom-Methode hatten wir $n = 3$ und $t = 1 \Rightarrow k \leq 3 - log_2 \left[\sum_{i=0}^{1} \binom{3}{i}\right] \Rightarrow k \leq 3 - log_2 \left[\binom{3}{0} + \binom{3}{1}\right] \Rightarrow k \leq 3 - log_2 [4]$ $\Rightarrow k \leq 1 \Rightarrow k = 1$. Wir sehen, dass wir dieses k in dem Beispiel benutzt haben.

3.3.1.3 ASWC mit der Paritäts-Methode

Wir stellten in Abschnitt 3.3.1.2 fest, dass die Nutzung der Syndrom-Decodierung optimal ist bei Slepian-Wolf-Codierung.

In der Praxis ist die statistische Korrelation zwischen der Quelle *X* und der zusätzlichen Information *Y* nicht im Voraus bekannt und variiert manchmal in der Zeit.

Somit müssen wir die Grenze der SW-Codierungskurve je nach aktueller $H(X/Y)$ korrigieren, in diesem Fall erlaubt ein Code mit der Fähigkeit zur Ratenanpassung, dass der Encoder die Codierungsrate während des Vorgangs ändert.

Dabei stellt sich die wichtige Frage, wie Codes mit anpassungsfähiger Codierungsrate für die SWC verwendet werden können?

Der systematische binäre lineare Code wird durch seine Generatormatrix G oder durch seine Paritäts-Prüfcode H definiert. Der Code ist gegeben durch

$$C = \{x: x \times H^T = 0\} \qquad 3.30$$

Das $\times$ in 3.30 ist das Multiplikationssymbol in Modulo 2, da der Code binär ist. Der Code sei (n, k). Wir können die Matrix *G* in eine andere Form überführen, diese ist

$$G = [I\ P] \qquad 3.31$$

Die Dimension der Matrix G ist gegeben durch $k \times n$, I ist die Einheitsmatrix mit der Dimension $k \times k$, P hat die Dimension $k \times (n - k)$. Das Code-Wort gehört zum Raum $\mathcal{F}_2^n$.

Durch Nutzung der Gleichung 3.31 können wir das Code-Wort darstellen

$$x = m \times G = [x_I\ x_P] \qquad 3.32$$

Das bedeutet, x_I ist die Originalnachricht m (von uns systematische Bits genannt) und x_P sind Zusatzbits zu den Originalbits, das bedeutet x_P sind die Paritätsbits des Codes.

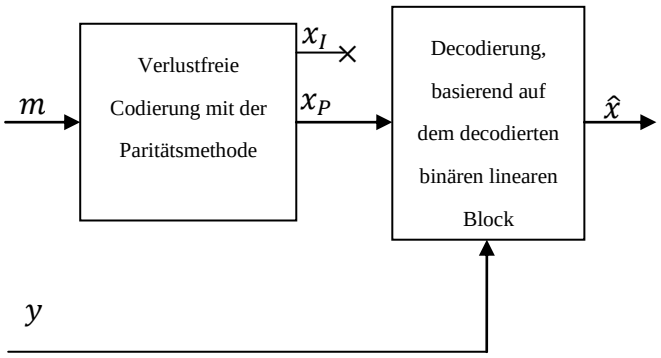


Abb. 3.10: Paritätsmethode für anpassungsfähigen code.

Abb. 3.10 zeigt die Paritäts-Methode. Die systematischen Bits x_I werden nicht übertragen, wir übertragen nur die Paritätsbits x_P. Das bedeutet, dass die Kompressionsrate bei dieser Methode $n/(n-k)$ beträgt.

Wir können hier auch die Zusatzinformation y als Rausch-Version der Nachricht m sehen. Da die Nachricht m und y binär sind, können wir uns vorstellen, dass m der Input des Rauschkanals BSC ist und y der Output, die Bitfehlerwahrscheinlichkeit P ist definiert durch die Korrelation zwischen der Quelle X und der Zusatzinformation Y.

In anderen Worten wird das Paar $[y\ x_P]$ als eine rauschbehaftete Version von $[x_I\ x_P]$ betrachtet, wir können auch sagen von $[m\ x_P]$.

Der Decoder eliminiert das Rauschen und triff die Vorhersage $\hat{x}$. Um dies ausführen zu können muss dem Decoder wie bei der Syndrom-Decodierung die Wahrscheinlichkeit des Fehlerbits im virtuellen Kanal bekannt sein und es müssen ausreichend Paritätsbits vom Encoder empfangen werden.

„ASWC mit der Syndrome-Methode“ ist äquivalent zur „ASWC mit der Paritäts-Methode“, nur wenn die Matrix H, die vom Encoder „ASWC mit der Syndrome-Methode“ verwendet wurde, zur Matrix P in der Gleichung 3.31 korrespondiert.

In [AG02] und [BM01] nutzten die Autoren den Turbo-Code für die Paritäts-Methode, in ihrem Schema erreichten sie eine Kompressionsrate von $n/2$.

Ein Problem bei der Nutzung der Paritäts-Methode ist auch, dass dem Encoder die Korrelation zwischen den Quellen X und Y bekannt sein muss, um sich für den passenden Code zu entscheiden, bzw. um festzustellen, wie viele Paritätsbits der Encoder X senden (das heißt, welche Generatormatrix G verwendet werden) muss.

Auf der anderen Seite muss diese Korrelation dem gemeinsamen Decoder bekannt sein, da für jede Kanal-Decodierung (wie Turbo-Code und LDPC-Code) zumindest die primäre Wahrscheinlichkeit zwischen den Originalbits m und der Zusatzinformation y bekannt sein muss.

Wie vorhin erwähnt, verändert sich bei realen Quellen die Korrelation zwischen X und Y von Zeit zu Zeit, was in solchen Fällen Methoden mit Möglichkeit zur Ratenanpassung notwendig macht. Die Anpassungsfähigkeit wird meist erzielt mittels eines Feedback-Kanals zwischen dem Encoder und dem gemeinsamen Decoder.

Dem Decoder muss die Qualität der Decodierung bekannt sein, um zwischen erfolgreicher und fehlerhafter Decodierung unterscheiden zu können. Im Falle Letzterer wird eine effiziente Vorhersagemethode gebraucht, um den prozentuellen Restfehler-Anteil zu bestimmen.

Die Bitfehlerrate (engl: bit error rate =BER) am Decoder-Output kann als Maß für die relative Qualität der Decodierung verwendet werden. Die Bitfehlerrate am Decoder kann durch Ermittlung der logarithmischen Wahrscheinlichkeitsverhältnisse (engl:log-likelihood ratio=LLR) berechnet werden, bspw. kann unter Nutzung von LDPC mit weicher Entscheidung (engl:LDPC soft decision) der Decoder die Bitfehlerrate vorhersagen. Wenn die Bitfehlerrate am Decoder einen vorgegebenen Wert übersteigt, wird der Decoder über einen Feedback-Kanal beim Encoder anfragen, mehr Paritätsbits zu senden.

Man kann für den Encoder vorsehen, dass die Sequenzen der Quelle X mit einer großen Menge Paritätsbits codiert werden und punktiert werden sollen (diese Punktierung ist durchgeführt bei Turbo oder LDPCA), dann soll an den Decoder die benötigte Menge Paritätsbits gesandt werden (normalerweise, wenn dem Decoder die Korrelation zwischen Quelle X und Zusatzinformation Y bekannt ist, kann dieser vorhersagen, wie viele Paritätsbits für eine fehlerfreie Decodierung benötigt werden).

Wenn die Korrelation für den Decoder optimal ist, also wenn $H(X/Y) \Rightarrow 0$ (in diesem Szenario sind keine Fehler in dem virtuellen Kanal vorgesehen), kann an der Ratengrenze in der Nähe der grünen Linien in Abb. 3.7 gearbeitet werden. Wenn die Korrelation sinkt (in anderen Worten wenn $H(X/Y) \Rightarrow H(X)$), entfernt sich die Ratengrenze von den grünen Linien in Abb. 3.7. Wenn keine oder nur falsche Information über die Korrelation dem Decoder zur Verfügung steht, kann sich die Ratengrenze weiter von den grünen Linien entfernen oder die Decodierung wird sogar unmöglich gemacht.

Wir sehen in Abb. 3.11, wie sich die Grenze der SW-Codierung verschlechtert mit abnehmender Korrelation zwischen den Quellen X und Y. Wir sehen, dass im ungünstigsten Fall für die beiden Quellen die Rate benötigt wird, die im Fall separater Codierung und Decodierung notwendig wäre.

Die Autoren in [TL05], [SCV04] entfernten die Syndrom-Bits des LDPC-Codes, was zur Verschlechterung der Leistung bei geringerer Rate führte.

In [AG02] und, [GFC04], [GFC04], [HM03] schlugen die Autoren einen anderen Weg der SW-Codierung mit der Paritäts-Methode vor, da es schwierig ist, das Syndrom zu punktieren. In [AG02] und [BM01] werden die vom Turbo-Encoder produzierten Paritätsbits in Puffer gespeichert, dann nach einem vorgegebenen Muster punktiert und auf Anfrage des Decorders über den Feedback-Kanal übertragen.

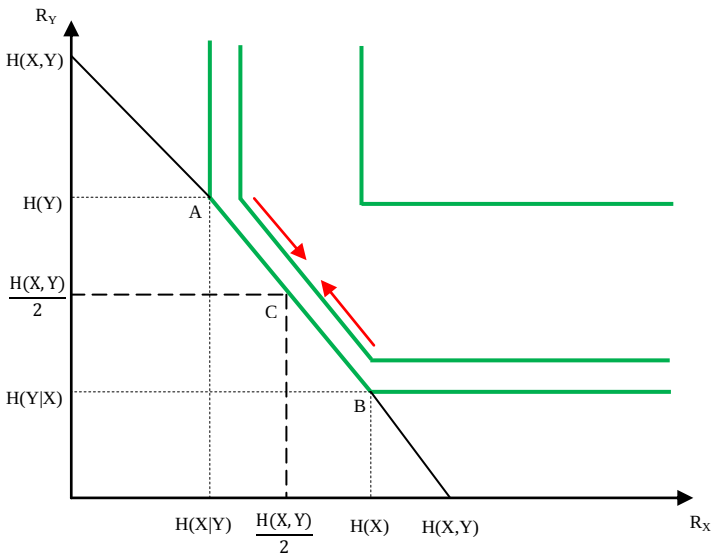


Abb. 3.11: Bereich der erreichbaren Raten von WZ für statistisch i.i.d. diskrete Zufallssequenzen X und Y, wenn die Korrelation zwischen den Qellen gerniger wird.

3.3.2 Nicht-asymmetrische Slepian-Wolf-Codierung

Bei der nicht-asymmetrischen SW-Codierung ist es erwünscht, dass die Raten der einzelnen Encoder variiert werden können, während die Gesamtsumme ihrer Raten $H(X,Y)$ konstant bleibt.

Das bedeutet, dass wir bei der nicht-asymmetrischen SW-Codierung an jedem Punkt zwischen A und B in Abb. 3.7 arbeiten können. Für einige Anwendungen ist die nicht-asymmetrische SW-Codierung besser geeignet als die asymmetrische.

Die symmetrische verteilte Quellencodierung bezieht sich auf den Fall, wenn beide Quellen bei der gleichen Kompressionsrate $R_X = R_Y$ komprimiert wurden. Das bedeutet, dass der Punkt c in Abb. 3.7 erreicht werden kann, allerding muss dieser dafür zwischen den Punkten A und B liegen, wofür folgende Gleichungen erfüllt sein müssen

$$\frac{H(X,Y)}{2} \geq H(X|Y) \quad 3.33$$

$$\frac{H(X,Y)}{2} \geq H(Y|X) \quad 3.34$$

$$\frac{H(X,Y)}{2} \leq H(X) \quad 3.35$$

$$\frac{H(X,Y)}{2} \leq H(Y) \quad 3.36$$

Eine der ersten Arbeiten, die sich mit der balancierten Zuweisung von Bitraten (engl:balanced rate allocation) bei der verteilten Codierung beschäftigt, basiert auf dem Zeitteilverfahren (engl:Timesharing).

Beim Zeitteilverfahren werden zwei asymmetrische Encoder und Decoder benutzt. Es gibt andere Methoden, bei denen jede Sequenz der Quellen mit einem linearen Kanalcode codiert wird, wobei nach der Codierung nur eine Fraktion der Informations-Bits und deren Paritätsbits gesandt werden (bei Nutzung der Paritäts-Methode) oder eine Fraktion Informations-Bits und deren Syndrom (bei Nutzung der Syndrom-Methode).

3.3.2.1 SW Ratengrenze beim Zeitteilverfahren

Wir nutzen das Zeitteilverfahren, um an jedem beliebigen Punkt zwischen A und B in Abb. 3.7 zu arbeiten.

Nehmen wir an, dass die Sequenzen von *X* und *Y* die Länge *n* haben und, wie vorher angenommen, dass der virtuelle Kanal zwischen *X* und *Y* ein BSC ist, wenn die Quellen binär sind.

Das Zeitteilverfahren hängt von dem Erreichen des Punktes B in Abb. 3.7 ab für eine Fraktion α von Sequenzen der Länge *n*, was bedeutet, dass wir diese Fraktion der Sequenz der Quelle *X* mit der Rate $R_{X_1} = H(X)$ codieren und die gleiche Fraktion der Sequenz der Quelle Y mit der Rate von $R_{Y_1} = H(Y/X)$. Wir erreichen dann den Punkt A für die Fraktion $1 - \alpha$ der Sequenz der zwei Quellen *X* und *Y*, was bedeutet, dass wir am Punkt A mit den Raten $R_{X_2} = H(X/Y)$ und $R_{Y_2} = H(Y)$ arbeiten. Daraus ergibt sich die folgende Gesamtbitmenge zum Erreichen eines jeden Punktes zwischen A und B

$$R_X = \alpha \mathrm{H(X)} + (1 - \alpha)\mathrm{H(X|Y)} \quad 3.37$$

$$R_Y = \alpha \mathrm{H(Y|X)} + (1-\alpha)\mathrm{H(Y)} \qquad 3.38$$

3.3.2.2 Nicht-asymmetrische Slepian-Wolf-Codeirung mit der Paritätsmethode

Wir nutzen in dieser Methode lineare Kanal-Codes wie folgt:

Wie in Abschnitt 3.3.1.3 wird jede Sequenz der Quellen *X* und *Y* der Länge *n* in zwei lineare Kanal-Encoder eingespeist.

Am Output des ersten linearen Kanal-Codes haben wir eine Sequenz der Länge $l + a$, wobei l die Länge der Output-Informations-Bits x_l der Sequenz *x* und *a* die Länge von deren Output-Paritätsbits P_a. Der Output des zweiten Kanalcodes hat eine Sequenz der Länge $(n-l)+b$, wobei $(n-l)$ die Länge der Output-Informations-Bits x_{n-l} der Sequenz *y* ist und b die Länge von deren Output-Paritätsbits P_b.

In diesem Szenario ist die Fraktion α gleich l/n, was bedeutet, dass um jeden Punkt zwischen A und B in Abb. 3.7 zu erreichen, wir die Bits bei folgenden Raten senden müssen

$$R_X \geq \frac{l}{\mathrm{n}}\mathrm{H(X)} + \frac{(\mathrm{n}-l)}{\mathrm{n}}\mathrm{H(X|Y)} \qquad 3.39$$

$$R_Y \geq \frac{(\mathrm{n}-l)}{\mathrm{n}}\mathrm{H(Y)} + \frac{l}{\mathrm{n}}\mathrm{H(Y|X)} \qquad 3.40$$

Der maximale Wert für l ist *n*, was für den Encoder von X bedeutet, dass keine Paritätsbits gesendet werden, in diesem Fall ist $\alpha = 1$. Wenn l Null ist, bedeutet dies, dass nur Paritätsbits gesandt werden, dann ist $\alpha = 0$.

Die erste Methode, um jeden Punkt innerhalb der SW-Grenze zu erreichen ist in [GFC04] beschrieben, in [GFC06] nutzte der Autor den Turbo-Code zur Punktierung, um nur α Bits für die Sequenz *x*, und *b* Bits für die Sequenz *y* übrig zu lassen.

Der Autor in [SF05] nutzte LDPC und das Zeitteilverfahren durchzuführen. In Seiner Arbeit bekommt der Empfänger einen Teil der Paritätsbits sowie die Original-Bits der Quelle *X* und der Decoder muss den zweiten Teil der Information aus Quelle *Y* als zu Sequenz *x* korrelierte Sequenz *y* nutzen. Der Decoder LDPC setzt voraus, dass der erste Teil der empfangenen Information fehlerfrei ist und der zweite Teil mit dem Bitfehlerwahrscheinlichkeit p (durch Nutzung des virtuellen BSC-Kanals) behaftet ist. Das bedeutet, der Decoder-LDPC für den ersten Teil mit folgenden Wahrscheinlichkeiten arbeitet

$$\Pr\left(x_{orig} = j \middle| x_{rec} = j\right) = 1, for\ j \in \{0,1\} \quad 3.41$$

In dieser Gleichung sind x_{orig} die Original-Bits des ersten Teils (welcher nicht nur Original-Bits, sondern auch Paritätsbits aufweist), x_{rec} sind die empfangenen Bits des ersten Teils.

Die Wahrscheinlichkeiten für den zweiten Teil sind

$$\Pr\left(y_{orig} = j \middle| y_{rec} = j\right) = 1 - p, for\ j \in \{0,1\} \quad 3.42$$

In dieser Gleichung sind y_{orig} die Bits des zweiten Teils (welcher nicht nur Original-Bits, sondern auch Paritätsbits aufweist), y_{rec} sind die empfangenen Bits des zweiten Teils.

3.3.2.3 Nicht-asymmetrische Slepian-Wolf-Codierung mit der Syndrom-Methode

Definition: Ein individueller SW-Code für die Codierung der Quellen $X_1, \dots, X_L$ bei den dazugehörigen Raten $R_1, \dots, R_L$, als Paar $(\mathsf{C}, \mathcal{M})$, wobei C ein (n, k) linearer Kanal-Code, gegeben durch die Generatormatrix $G_{k\times n}$ und $\mathcal{M}$ eine geordnete Menge ganzer Zahlen $\{m_1, \dots, m_L\}$ ist, so dass $\sum_{j=1}^{L} m_j = k$,wo $m_j = n - nR_j$ und R_j die Kompressionsrate für X_j ist.

Diese Methode ist optimal (in dem Sinne, dass die Slepian-Wolf-Grenze für multiple Quellen erreicht wird), wenn der Code C die Kapazität des äquivalenten Korrelationskanals zwischen den Quellen erreicht.

Nehmen wir einen Fall an, in dem wir nur zwei Quellen haben, was bedeutet, dass L=2. Wir formen den Code C_1 als Sub-Code des Hauptcodes C mit $m_1 \times n$ Generatormatrix G_1, die aus m_1 Reihen von $G_{k\times n}$ besteht und formen den Code C_2 als Sub-Code des Hauptcodes C mit einer $m_2 \times n$ Generatormatrix G_2, die aus m_2 Reihen von $G_{k\times n}$ besteht. Dabei ist $m_1 + m_2 = k$.

Die Wahl der Reihen m_1 ist irrelevant, solange sie linear unabhängig sind von denen, die von einem anderen Sub-Code besetzt sind.

Die Anzahl von m_1 und m_2 erlaubt die Auswahl jeder Rate zwischen den Punkten A und B der Slepian-Wolf-Grenze in Abb. 3.7.

Die Paritäts-Matrices für die Codes C_1 und C_2 sind H_1 der Größe $(n - m_1) \times n$, und H_2 der Größe $(n - m_2) \times n$.

Der Encoder sendet das Syndrom der Sequenzen unter Nutzung der Paritäts-Prüfmatrix, was bedeutet, dass der Encoder von X das Syndrom $S_X = x.H_1^T$ der Länge $n - m_1$ sendet und der Encoder von Y das Syndrom $S_Y = y.H_2^T$ der Länge $n - m_2$. Die Gesamtrate zur Codierung der Input-Sequenzen beträgt

$$R_1 + R_2 = 2n - (m_1 + m_2) = 2n - k. \qquad 3.43$$

Die Nebenklassen, die (x, y) enthalten, sind veränderte Versionen der Codes C_1 und C_2 mit den Fehlervektoren t_1 und t_2.

Wir nehmen an, $a_1 G_1$ sei ein Code-Wort in C_1 und $a_2 G_2$ ein Code-Wort in C_2, wo a_1 ist eine Sequenz in Quelle X_1, und a_2 ist eine Sequenz in Quelle X_2. Was bedeutet, dass das Syndrome dieser Code-Wörter Null sind. Nehmen wir auch an, *S* ist das Syndrom des Codes C. In diesem Fall findet der Decoder alle Code-Wörter in einer Nebenklasse des Syndroms *S*, wenn er ein zum Null-Syndrom gehörendes Code-Wort mit dem zur Nebenklasse des Syndroms *S* gehörenden Fehlervektor *t* verschiebt.

Wir können also sagen, dass es sich bei $x \oplus t_1$und $y \oplus t_1$ um Code-Wörter der Codes C_1 beziehungsweise C_2 handelt.

Wenn der Decoder S_X und S_Y von den zwei Encodern empfängt, findet der gemeinsame Decoder das Code-Wort c in C welches nah (in Hamming-Distanz) an $t = t_1 + t_2$ ist. Wir wissen auch, dass das Syndrom des Vektors *t S* ist. Wir suchen nach dem Code-Wort, welches zu dem gleichen Sydrom *S* gehört, aber das minimale Gewicht aufweist (engl:the minimum weight). Dies bedeutet, wir suchen das minimale Fehlermuster, das zum gleichen Syndrom *S* gehört.

Wenn die maximale Hamming-Distanz zwischen den Sequenzen der Quellen *X* und *Y* $d_{maxH}(x, y)$ beträgt, benötigen wir zur Nutzung des Codes C bei fehlerfreier Decodierung

$$d_{minH}(C) \geq 2d_{maxH}(x, y) + 1 \qquad 3.44$$

Die erste Arbeit über diese Syndrom-Methode liegt in [PR00a] vor. In [SPR03] wandte der Autor diese Methode auf nicht-gleichverteilte Quellen an. In [SLXG06] nutzte der Autor die Methode für nicht-systematische lineare Codes.

In [GD05] wurde diese Methode genutzt, um einen Teil der Quellen-Informationsbits zu senden.

3.4 Wyner-Ziv-Codierung

Wyner-Codierung generalisiert den Aufbau der Slepian-Wolf-Codierung, so dass bei der Codierung von *X* das Genauigkeitskriterium (engl:fidelity criterion) eingeführt wird anstatt wie vorher eine verlustfreie Codierung auszuführen. Dieses Problem der verlustbehafteten verteilten Codierung wurde von Wyner and Ziv in [WZ76] vorgestellt.

Angenommen *Y* ist direkt verfügbar für den Decoder, in dem Fall muss das System so konstruiert sein, dass *X* bei der optimalen Rate codiert wird, so dass bei der Wiederherstellung von *X* am Decoder die Verzerrung in einem bestimmten Rahmen bleibt.

Wyner und Ziv berechneten die für den Decoder optimale Rate $R^{WZ}(\mathrm{D})$ zur Übertragung von *X*, so dass der Decoder bei der Wiederherstellung von $\hat{X}$ innerhalb einer durchschnittlichen Verzerrung D von *X* ausführen kann (das spezifische Maß für eine Verzerrung kann z. B. der mittlere quadratische Fehler sein).

Definition:
Bei Vorgabe einer diskreten gedächtnislosen Quelle *X*, einer diskreten gedächtnislosen Quelle *Y* als Zusatzinformation, und der unabhängigen, identisch verteilten (engl: identical independently distributed =i.i.d) Sequenzpaare (x(k), y(k)) über k und der Verzerrungsfunktion $d: X \times U \longrightarrow \mathcal{R}^{+}$, kann eine Rate R erzielbar werden bei verlustbehafteter Quellencodierung mit Zusatzinformation am Decoder und mit Verzerrung D, wenn $\mathrm{R} > \mathrm{R}^{\mathrm{WZ}}(\mathrm{D})$.

$$\mathrm{R}^{\mathrm{WZ}}(\mathrm{D}) = \min_{\substack{\mathrm{p(u|x)} \\ \mathrm{U-X-Y} \\ \mathrm{E[d(X,U)]\leq D}}} \mathrm{I(X;U|Y)} \qquad 3.45$$

Der Parameter $\mathrm{R}^{\mathrm{WZ}}(\mathrm{D})$ wird als Rateverzerrung-funktion mit der Zusatzinformation Y bezeichnet. Das Konzept eines solchen Systems wird als das Problem der verlustbehafteten Quellencodierung (engl: loss source coding) mit Zusatzinformation bezeichnet. Ist die Rate R kleiner als $\mathrm{R}^{\mathrm{WZ}}(\mathrm{D})$, kann nicht bei Rate R gearbeitet werden.

Das Wyner-Ziv-Theorem stell dar, dass wenn die statistische Abhängigkeit zwischen Quelle *X* und Zusatzinformation *Y* nur am Decoder ausgenutzt wird (wie SW-Coding), die Übertragungsrate sich erhöht im Vergleich zu dem Fall, wenn die Korrelation am Encoder und am Decoder ausgenutzt wird (der Fall gemeinsam Encoding und Decoding). Dies gilt, wenn die durchschnittliche Verzerrung D die gleiche ist. Mathematisch beschreiben lässt sich dies durch

$$\mathrm{R}^{WZ}(\mathrm{D}) \geq \mathrm{R}_{\mathrm{X|Y}}(\mathrm{D}) \qquad 3.46$$

Wyner-Ziv beweist auch, dass

$$\mathrm{R}^{WZ}(\mathrm{D}) \leq \mathrm{R}_{\mathrm{X}}(\mathrm{D}) \qquad 3.47$$

Wobei $\mathrm{R}_{\mathrm{X|Y}}(\mathrm{D})$ die minimale Rate darstellt, bei der *X* codiert werden kann, wenn *Y* am Encoder und am Decoder verfügbar ist, $\mathrm{R}^{WZ}(\mathrm{D})$ stellt die minimale Wyner-Ziv-Rate dar, die zur Codierung von *X* nötig ist, wenn *Y* nur am Decoder zur Verfügung steht. $\mathrm{R}_{\mathrm{X}}(\mathrm{D})$ ist schließlich die Rateverzerrung-funktion für separate Codierung, und Decodierung der Quelle *X*.

Man erkennt aus der Gleichung 3.46 , dass wenn die Verzerrung Null ($\mathrm{D} = 0$) ist, es zu keinem Ratenverlust kommt, falls die zusätzliche Information nur am Decoder verfügbar ist. Wir werden sehen im Abschnitt 3.4.2, dass bei Gleichung 3.46 dann Gleichheit auftritt, wenn Quelle *X* und Zusatzinformation *Y* gemeinsam eine Gaußsche Verteilung aufweisen bei Angabe der Verzerrung D durch den mittleren quadratischen Fehler. In der Literatur wird $\mathrm{R}_{\mathrm{X|Y}}(\mathrm{D})$ als Rate-Verzerrung-funktion bezeichnet.

3.4.1 Wyner-Ziv-Coderuing für den binären symmetrischen Fall

Es kann damit gearbeitet werden wie im Falle der SW-Codierung, bei der SWC wurde ein linearer Block-Code (n_{old}, k_{old}) eine Menge von $2^{k_{old}}$ binären Wörtern der Länge n_{old} zu indizieren unter Nutzung von $2^{n_{old}-k_{old}}$ Teilmengen. Jede Teilmenge ist indiziert mit einem eindeutigen Syndrom. Die Anzahl des Syndroms ist $n_{old} - k_{old}$.

Dieser Code wurde konzipiert für Sequenzen *x* der Quelle *X* von der Länge n_{old} und hat eine minimale Hamming-Distanz länger oder gleich $2d_{maxH}(x, y) + 1$, wobei d_{maxH} is die maximale Hamming-Distanz zwischen den Sequenzen der Quelle *X* und den Sequenzen der Zusatzinformation *Y* ist.

Zur Ausführung der Wyner-Ziv-Codierung müssen wir nicht alle Teilmengen des Codes nutzen, brauchen aber eine kleinere Anzahl von Syndromen, um die binären Wörter zu indizieren, was bedeutet, dass bei dem neuen Code (n_{new}, k_{new}) die minimale Hamming-Distanz des Codes (n_{old}, k_{old}) nicht erhalten bleibt. Die minimale Hamming-Distanz des neuen Codes wird kleiner sein als die Hamming Distanz des Codes (n_{old}, k_{old}). Dies führt zu einer nicht-fehlerfreien Wiederherstellung der Sequenzen *x* bei verfügbaren Sequenzen *y* und Empfang des Syndrom am Decoder.

Um eine kleinere Syndrom-Anzahl für den neuen Code zu erzielen, müssen wir entscheiden, dass

$$\mathrm{n_{new}} - \mathrm{k_{new}} = \mathrm{k_1} - \mathrm{k_{old}}, \mathrm{n_{old}} > k_1 > \mathrm{k_{old}} \qquad 3.48$$

Es ergibt sich aus der Gleichung 3.48, dass der neue Code $2^{k_1-k_{old}}$ Syndrome nutzt, während der alte Code $2^{n_{old}-k_{old}}$ Syndrome brauchte, was bedeutet, dass bei der WZ-Codierung die binären Code-Wörter des alten Codes zu den nächstliegenden zulässigen Code-Wörtern des neuen Codes quantisiert werden müssen.

Die Anzahl der zulässigen Code-Wörter im neuen Code ist gegeben durch die Zahl des Syndroms des neuen Codes multipliziert mit der Anzahl der Codewörter, die zu einem Syndrom gehören. Mathematisch lässt es sich beschreiben durch $2^{k_1-k_{old}} \times 2^{k_{old}} = 2^{k_1}$.

Dieser Quantisierungsprozess kann als ein linearer Block-Code (n_{old}, k_1) gesehen werden, wir betrachten diesen Code als einen linearen Block-Quellen-Code im Vergleich zum linearen Block-Code (n_{old}, k_{old}), da der lineare Block-Code (n_{old}, k_1) jedem Syndrom eine größere Anzahl der Elemente $2^{k_1} > 2^{k_{old}}$ zuweist als der Code (n_{old}, k_{old}), das führt zu einer geringeren Anzahl der Syndrome $2^{n_{old}-k_{old}} < 2^{n_{old}-k_1}$ als bei dem Code (n_{old}, k_{old}).

Für die gleiche Diskussion wie oben betrachten wir den linearen Block-Code (n_{old}, k_{old}) als eine grobe Kanal-Codierung, die in den endgültigen, feinen Kanal-Quellen-Code (n_{old}, k_1) geschachtelt ist. Wir können die Wyner-Ziv-Grenze erreichen (siehe Abb. 3.12), indem wir das Verlustproblem unterteilen in Ratenverluste bei der Quellencodierung im Zusammenhang mit dem linearen Block-Code (n_{old}, k_1) und Ratenverluste bei der Kanalcodierung in Zusammenhang mit dem linearen Block-Code (n_{old}, k_{old}).

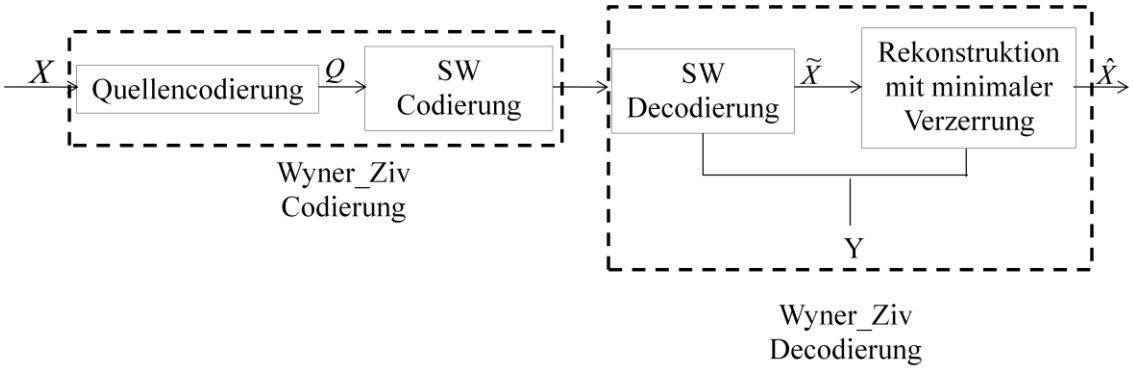


Abb. 3.12: Blockdiagramm einer allgemeinen Wyner-Ziv Codierung.

Das bedeutet, dass wir zur Durchführung einer guten Wyner-Ziv-Codierung einen guten feinen Quellencode und einen guten groben Kanalcode benötigen.

Der Autor in [TT91] betrachtete den Fall, wenn die Quellen *X* und *Y* binär symmetrisch sind, die Korrelation zwischen ihnen als BSC mit Änderungswahrscheinlichkeit *p* dargestellt ist und das Maß für die Verzerrung zwischen der Sequenz *x* und deren rekonstruierter Sequenz ist die Hamming-Distanz. Er berechnet die Rateverzerrung-

funktion $R_{X|Y}(D)$ (sieh Gleichung 3.46) der verlustbehafteten Codierung von X, im Falle dass Y dem Encoder und dem Decoder zur Verfügung steht.

Die Rate ist dann gegeben durch

$$R_{X|Y}(D) = \begin{cases} H(p) - H(D), & 0 \leq D \leq \min\{p, 1-p\} \\ 0 \quad , & D > min\{p, 1-p\} \end{cases} \qquad 3.49$$

In [WZ76], [ZSE02] beschäftigte der Autor sich mit der Rate-Verzerrung-Funktion $R^{WZ}(D)$ im Falle, dass

$$R^{WZ}(D) = l.c.e\left\{H\big((1-p)D + (1-D)p\big) - H(D)\,, (p, 0)\right\} \qquad 3.50$$

Das gilt für $0 \leq D \leq p$. $l.c.e$ bedeutet untere konvexe Hülle (engl:the lower convex envelope).

Für $p \leq 0.5$ haben wir $R^{WZ}(D) \geq R_{X|Y}(D)$, mit Gleichheit nur an zwei Rateverzerrung-Punkten (siehe Abb. 3.13). Der erste Punkt ist der Raten-Nullpunkt am Index $(p, 0)$ und der zweite Punkt ist der Verzerrungs-Nullpunkt $(0, H(p))$, dieser Punkt korrespondiert mit dem Slepian-Wolf-Punkt, da dort kein Verlust entsteht, laut Gleichung 3.49 ist es äquivalent zu dem Fall, dass

$$R^{WZ}(0) = R_{X|Y}(0) = H(X|Y) = H(p) \qquad 3.51$$

Für $p > 0.5$ haben wir $R^{WZ}(D) > R_{X|Y}(D)$, was bedeutet, dass wir bei der WZ-Codierung immer Verluste haben, wenn die zusätzliche Information nur am Decoder zur Verfügung steht.

Der Autor in [LXG03] zeigt $R^{WZ}(D)$ und $R_{X|Y}(D)$ für den binären symmetrischen Fall bei $p = 0.27$ (sieh Abb. 3.13). Wir sehen, dass in diesem Fall die maximale Differenz zwischen $R_{X|Y}(D)$ und $R^{WZ}(D)$ kleiner ist als 0.1 Bit (nur für den Fall, dass $p = 0.27$). Wir sehen in der Abbildung auch das Zeitteilverfahren zwischen dem Raten-Nullpunkt und dem Verzerrungs-Nullpunkt, all die Punkte des Zeitteilverfahrens weisen das gleiche Verhältnis $H(p)/p$, also Bits/Verzerrung, auf.

Die Abbildung zeigt auch die Ergebnisse der Simulation, bei der versucht wurde, die theoretische Grenze der WZ-Codierung zu erreichen, wir sehen, dass die kleinste Differenz zwischen der theoretischen Grenze und den Simulationspunkten 0,09 Bit betragen kann.

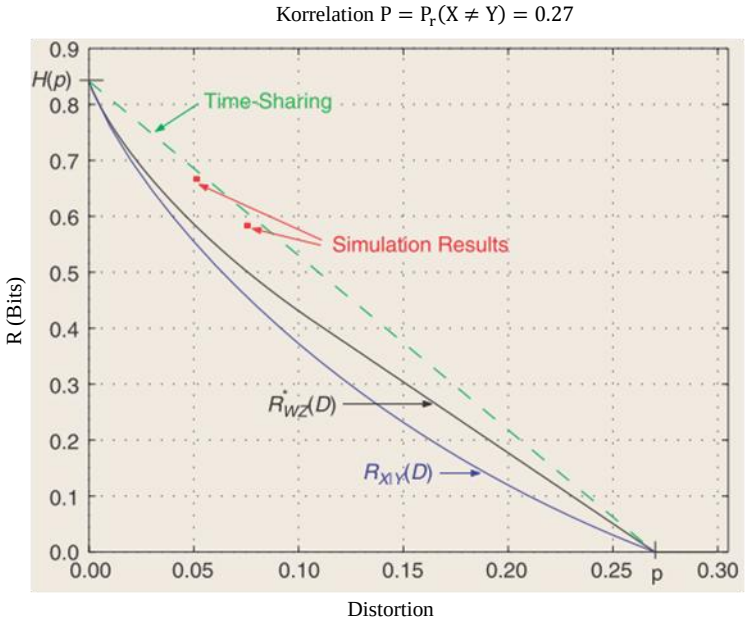


Abb. 3.13: Simulierte Leistung des verschachtelten Schemas aus [LXG03] für die binäre Wyner-Ziv-Codierung bei einer Korrelation P = 0.27. Die Zeitteillinie (engl: Time-Scharing) zwischen dem Null-Raten-Punkt (P, 0) und dem Slepian-Wolf-Punkt (0, H(P)) wird auch gezeigt.Quelle: [LXG03].

In [ZSE02] zeigte Zamir, dass im Falle allgemeiner Statistiken und gedächtnisloser Quellen mit endlicher Varianz sind, indem er den mittleren quadratischen Fehler als Maß für die Ratenverzerrung zu Hilfe nahm, in diesem Fall ist $R^{WZ}(D)$ größer als $R_{X|Y}(D)$, maximal 0,5 Bit/Quellensymbol

$$R^{WZ}(D) - R_{X|Y}(D) \leq 0.5 \text{ bit/source} \qquad 3.52$$

Die Idee von Zamir wurde weiterentwickelt und angewandt von Pradhan et al. Die Autoren in [KDR01], [PKR02], [PR99] und Servetto [Ser00] konzentrieren sich auf verschachtelt skalierte Quantisierung. Der Autor in [Llo82] nutzte den Lloyd-Algorithmus für die WZ-Codierung.

In [FZE04], [ME02], [CVA02] beschäftigten sich die Autoren mit anderen Fällen, um Wyner-Ziv-Codierung und die Quantisierung weiterzuentwickeln.

3.4.2 Wyner-Ziv-Codierung für den Quadratischen Gaußschen Fall

Die Autoren in [WTV08] beschäftigten sich mit der Wyner-Ziv-Codierung für den Quadratischen Gauss´schen Fall.

Wenn Quelle *X* und Zusatzinformation *Y* beide stationär und gedächtnislos sind sowie die Gaußsche Verteilung mit dem Mittelwert Null aufweisen und i.i.d. in k sind, ist (X(k), Y(k)) die Gaußsche Verteilung mit dem Mittelwert Null, $(X(k), Y(k)) \sim \mathcal{N}(0, \Lambda)$.

Und wenn die Verzerrungsfunktion *D* der mittlere quadratische Fehler ist, ist die Kovarianz-Matrix Λ von X und Y, gegeben durch

$$\Lambda = \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix} \qquad 3.53$$

Wenn $|\rho| < 1$, entsteht bei Nutzung der WZ-Codierung im quadratischen Gaußschen Fall kein Ratenverlust im Vergleich zu dem Fall, wenn die Quelle *Y* am Encoder und am Decoder verfügbar ist.

Für die Rate-Verzerrungs-Funktion $d(x,u) = (x-u)^2$, ergibt sich in diesem Fall

$$R^{WZ}(D) = R_{X|Y}(D) = \frac{1}{2}\log^+\left[\frac{\sigma_X^2(1-\rho^2)}{D}\right] \qquad 3.54$$

Wobei $\log^+ x = \max\{\log_2 x, 0\}$.

Dieses Ergebnis für den Gaußschen Fall wird verwendet bei der verteilten Video-Codierung, wenn zum Beispiel die geraden und ungeraden Frame-Sequenzen zusammen eine Gaußsche Verteilung ergeben.

3.4.3 Praktische Wyner-Ziv-Codierung mittels verschachtelt skalierter Quantisierung

Wir haben gesehen, dass WZC zu Verlusten im Vergleich zu SWC führt, das führt dazu, dass die Quantisierung als Quellencodierung für Quelle X eingesetzt wird in Abb. 3.12. Nach dieser Quantisierung bleibt noch eine Korrelation zwischen der quantisierten Quelle *X* und deren Zusatzinformation *Y*.

In Abb. 3.14 sieht man das praktische Diagramm für die WZ-Codierung bei Nutzung der Quantisierung.

Wir konstruieren die Quantisierung in Abhängigkeit von der Verzerrung zwischen x und $\hat{x}$. Nach diesem Quantisierungsprozess nutzen wir die SW-Codierung um die Rate der quantisierten Quelle *X* zu verringern, indem wir die Korrelation zwischen *X* und *Y* ausnutzen.

Wir wissen aus Abschnitt 3.1, dass die SW-Codierung auf Kanalcodierung basiert, daher können wir sagen, dass die WZ-Codierung ein Quellen-Kanal-Codierungsproblem ist.

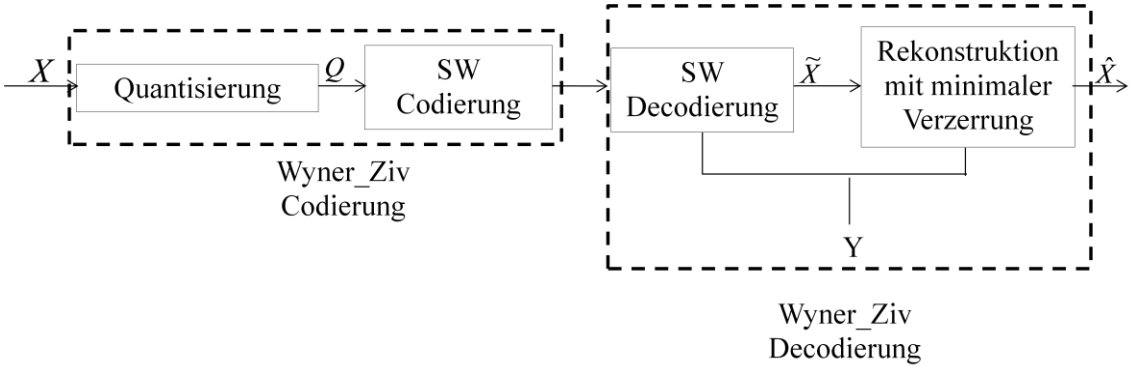


Abb. 3.14: Blockdiagramm der praktischen Wyner-Ziv-Codierung.

Bei der WZ-Codierung treten nicht nur Verluste auf, die auf Quantisierung bei der Quellencodierung zurückzuführen sind, sondern auch bei solche, die bei der Kanalcodierung durch die Unterteilung in Teilmengen (engl:binning) entstehen. Daher wird zum Erreichen der WZ-Grenze sowohl eine gute Quellencodierung als auch eine gute Kanalcodierung benötigt. Der Decoder nutzt die zusätzliche Information Y, um die Verzerrung des Signals $\hat{x}$ zu reduzieren.

Nachdem der Encoder die Sequenz x quantisiert hat (was zu Quantisierungsverlusten führt), sendet der Encoder den Index zur Identifizierung der Teilmenge, die das quantisierte x enthält. Diese Quantisierung hängt ab von der verschachtelt skalierten Quantisierung.

Verschachtelt skalierte Quantisierung:

Verschachtele skalierte Quantisierung ist der skalaren Quantisierung ähnlich, außer dass jede Teilmenge (engl:bin) aus Sub-Teilmengen (engl:sub-bin) besteht, und die Teilmengen über den Bereich der Quelle X gleich verteilt sind.

Wenn a und b die Indices der niedrigsten und höchsten Quantisierungsebenen darstellen, ist die Anzahl der Sub-Teilmengen N (engl:sub-bin) in jeder Teilmenge gegeben durch

$$N = a + b + 1 \tag{3.55}$$

Die Sub-Teilmengen jeder Teilmenge weisen jeweils die Weite q auf, jede Teilmenge hat dann die Weite Nq.

Das quantisierte Code-Wort für die Sub-Teilmenge i ist gegeben durch

$$\tilde{x} = iq, i = a, a + 1, \dots, b \tag{3.56}$$

Die Grenzen zwischen den Sub-Teilmengen sind gegeben durch

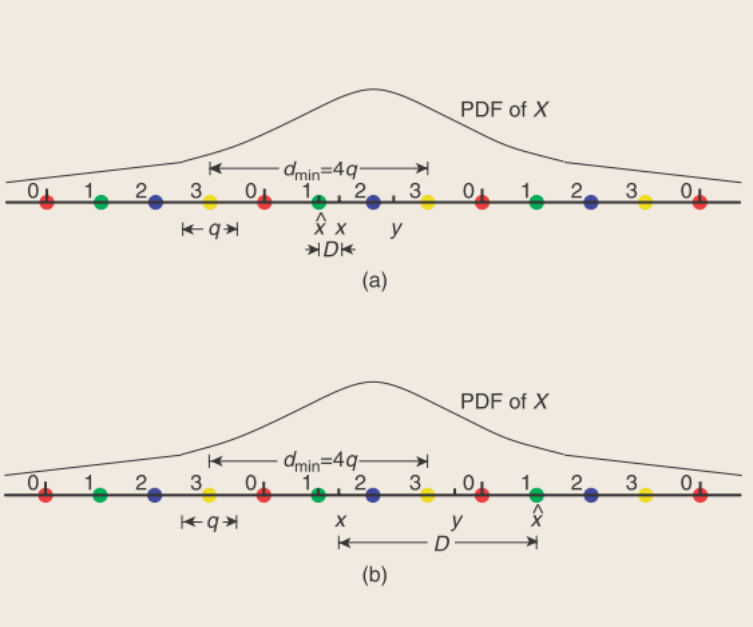


Abb. 3.15: 1-D verschatelte Lattice gleichförmige Quantisierung mit vier Teilmengen (engl:bins) für das Quadratische Gaußsche Wyner-Ziv-Problem, wobei die Zusatzinformation Y nur am Decoder zur Verfügung steht. (a) "nur gute" Verzerrung. (b) "gute" und "schlechte" Verzerrung.

$$t_i = (i - 0.5)q, i = 0, \mp 1, \mp 2, \ldots \qquad 3.57$$

Beispeil einer Wyner-Ziv-Codierung mittels verschachtelt skalierter Quantisierung

Wir gehen von zwei Gaußchen gedächtnislosen Quellen X und Y (sieh Abb. 3.15). Wir legen den Quantisierungsparameter q fest in Abhängigkeit von der Verzerrung zwischen x und $\hat{x}$, das Ergebnis sollte sein $q \leq d(x, \hat{x})$.

In Abhängigkeit von Parameter q und der Korrelation zwischen der Zusatzinformation und der quantisierten Sequenz $\hat{x}$ legen wir fest, wieviele Sub-Teilmengen in der Teilmenge sein sollen, um N festzulegen.

In Abb. 3.15 sehen wir, dass wir $N = 4$ und $q = 1$ setzen, das bedeutet, wir haben 4 Sub-Teilmengen, a ist Null in 3.55 und b ist 3. Jede Sub-Teilmenge hat die Weite $q = 1$, jede Teilmenge hat die Weite $Nq = 4q = 4$.

In Abhängigkeit von der Gleichung 3.56 ist das Quantisierte Code-Wort für die Teilmenge i gegeben durch

$i = a = 0 \Rightarrow \tilde{x} = iq = 0$

$i = a + 1 = 1 \Rightarrow \tilde{x} = iq = 1$

$i = a + 2 = 2 \Rightarrow \tilde{x} = iq = 2$

$i = b = a + 3 = 3 \Rightarrow \tilde{x} = iq = 3$

Der Quantisierungsprozess rundet die Sequenz *x* auf ihre nächste Quantisierungsebene x_q, dann sendet der SW-Encoder nur den Index des Quantisierungs-Code-Wortes $\tilde{x}$. Der SW-Encoder schafft dazu Teilmengen, jeden von denen enthält die Intervalle aller Sequenzen *x* mit dem gleichen Quantisierungs-Code-Wort $\tilde{x}$. In unserem Beispiel nutzt der Encoder 4 Teilmengen, anschließend sendet der Encoder nur das $\tilde{x}$ der Teilmenge, zu der die quantisierte Sequenz x_q gehört, für unser Beispiel bedeutet es, dass der SW-Encoder nur 2 Bits für jede Sequenz *x* der Quelle *X* braucht.

Der SW-Decoder empfängt den Index $\tilde{x}$, kann ihn aber noch keiner präzise Teilmenge zuordnen. In Abhängigkeit von der Zusatzinformation *y* berechnet der Decoder die Teilmenge der Sequenz *y* unter der Annahme, dass es die gleiche Teilmenge ist, zu der Sequenz *x* gehört. Schließlich findet der Decoder an Hand der wiederhergestellten Teilmenge und des empfangenen Index $\tilde{x}$ den Intervall-Wert der Sequenz *x*.

Der Minimalverzerrungs-Rekonstruktionsblock in Abb. 3.14 kann mit Hilfe der Korrelation zwischen Sequenz x_q und Zusatzinformation *y* die Wiederherstellung von x_q dahingehend ändern, dass die maximale Nähe (im Sinne der Hamming-Distanz) zu *y* hergestellt wird.

Das Problem tritt auf, wenn die Zusatzinformation *y* im Grenzbereich der Teilmenge liegt (sieh Abb. 3.15 (b)), was dazu führt, dass der Decoder $\tilde{x}$ zur falschen Teilmenge zuordnet, so dass die Distanz zwischen dem tatsächlichen Wert von x_q und dessen rekonstruiertem Wert groß wird, in Abb. 3.15 (b) z.B. wird die Distanz (man kann sagen die Verzerrung *D*) größer als $3q = 3$.

Wir sehen, dass das Problem vermieden werden kann, indem die Weite der Teilmengen vergrößert wird. Das führt zu einer geringeren Wahrscheinlichkeit, dass die Sequenz *y* am Rand einer Teilmenge oder in der falschen Teilmenge liegt.

Wenn wir Nq vergrößern, führt dies zu einem höheren Wert für q bei konstantem N ("konstantes N" bedeutet, dass die Rate R_X konstant ist, so dass wir in unserem Beispiel nur 2 Bits für jede Sequenz *x* senden). Wenn wir N konstant und q größer setzen, führt dies zu einer größeren Verzerrung zwischen *x* und deren quantisiertem Wert x_q am Decoder bei der SW-Codierung.

Vice versa, wenn man q verkleinert ohne N zu verändern, verringert sich die Verzerrung während des Quantisierungsprozesses, und die Verzerrung im Minimalverzerrungs-Rekonstruktionsblock in Abb. 3.15 wird dagegen größer.

Für ein festes N existiert daher ein optimales q, welches die Gesamtverzerrung minimiert.

Wir sehen, dass in der Praxis zur Durchführung einer guten WZ-Codierung sowohl ein guter Quantisierungsprozess (gute Quellencodierung) als auch eine gute SW-Codierung (gute Kanalcodierung) benötigt werden.

Wenn wir sicherstellen wollen, dass die Teilmenge der Zusatzinformation y nicht falsch ist, muss die maximale Distanz zur quantisierten Sequenz x_q gegeben sein durch

$$\frac{q}{2} > d_{maxH}(x_q, y) \qquad 3.58$$

Allerdings wird dies wieder zur gleichen Diskussion um die Optimierung des Parameters q führen, wie weiter oben.

Der Autor von [CX04] beschäftigte sich mit praktischer WZ-Codierung für Gaußsche Quellen, die auf verschachtelt skalierter Quantisierung basiert und nutzte dabei LDPC zur SW-Codierung.

3.5 Simulationsergebnisse für die praktische SW/WZ bei Nutzung von Kanalcodierung

Wir gehen von zwei binären gleichverteilten Quellen *X* und *Y* aus, wir nehmen an, dass die Korrelation zwischen Quelle *X* und deren Zusatzinformation durch einen virtuellen BSC gegeben ist mit der Änderungswahrscheinlichkeit $p = 0.11$ was zur bedingten Entropyrate $R_{X|Y} = H(X|Y) = H(p) = 0.5$ Bit führt, dies bedeutet, dass die Kompressionsrate für die Codierung der Quelle *X* 2:1 ist, das darauf zurückzuführen, dass $\frac{H(p)}{H(X)} = \frac{0.5}{1} = 0.5$ ist. Wir nehmen auch an, dass wir die SW-Grenze erreichen, wenn die Fehlerwahrscheinlichkeit für die Decodierung von *X* kleiner oder gleich 10^{-6} ist.

In Abhängigkeit vom virtuellen BSC-Kanal steigt mit Verringerung der Wahrscheinlichkeit p die Korrelation zwischen Quelle *X* und Zusatzinformation *Y*, was zur Verkleinerung von $H(p)$ führt. In Abb. 3.16 ist der Wert der Korrelation auf der horizontalen Achse, und die Fehlerwahrscheinlichkeit für das decodierte *X* auf der vertikalen Achse dargestellt. Die Leistungsgrenze der WZ-Codierung ist ebenfalls dargestellt.

Es gibt zwei SW-Codierungsmethoden, mit Turbo-Code und LDPC-Code, die Code-Wort-Längen der SWC-Methoden betragen jeweils N=10^4 ,10^5. Aber beide haben Kompressionsraten von 2:1 bei Kanal-Codierung.

Wir sehen, dass SW bei Nutzung gleicher Kanalcodierung (Turbo oder LDPC) bei größerer Code-Wortlänge *N* leistungsstärker wird. Es ist zu sehen, dass je größer die Korrelation ($H(p)$ kleiner), desto geringer wird die Fehlerwahrscheinlichkeit für einen

bestimmten SW-Code. Je leistungsstärker der SW-Code, desto geringere Korrelation (größeres $H(p)$) braucht der Code zum Erreichen einer sehr kleinen Fehlerwahrscheinlichkeit.

Wir sehen auch, dass der reguläre LDPC ein wenig leistungsstärker ist als die Turbo-Codierung, die mit größerer oder gleicher Code-Wortlänge N arbeitet, während der irreguläre LDPC den Abstand zur SW-Grenze beinahe halbiert (dies trifft zu bei N=10^4 und N=10^5) .

Wir sehen auch, dass die beste Leistung für einen nicht-syndromischen Turbo-Code bei ca. 0,38 Bit für eine Code-Wortlänge N=10^5 liegt.

Laut Abb. 3.16 erweisen sich die Turbo-Codes als leistungsstärker im Vergleich zu den regulären, aber nicht zu den irregulären LDPC-Codes. Wir sehen auch, dass der Turbocode bei Nutzung von Parallelverkettung (engl.:parallel concatenation) mit N=10^5 schneller konvergieren kann als der Turbo-Code, der serielle Verkettung nutzt (dies trifft sowohl zu, wenn N=10^4 als auch bei N=10^5).

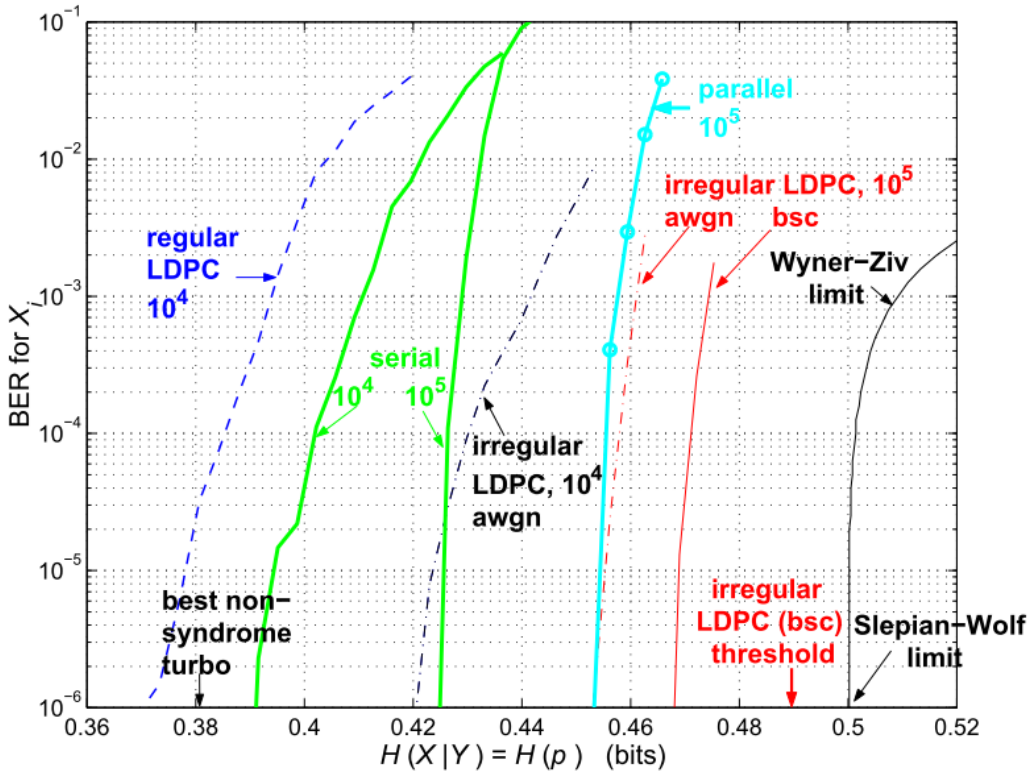


Abb. 3.16: Die theoretischen Grenzen für WZ/SW-Codierung im Vergleich mit regulären und irregulären LDPC- und Turbo-Codes. Quelle: [LXG02].

4 Niedrigdichten Paritätsprüfcode (Low density parity check code=LDPC)

In diesem Kapitel beschäftigen wir uns mit der LDPC-Kanalcodierung, mit deren Eigenschaften und zwei wihtigen Arten dieser Kanalcodierung. Wir werden die Komplexität von Codierung und Decodierung berechnen und uns im Detail mit der Konstruktion der Generator- und der Decodierungsmatrix des LDPC befassen. Danach werden wir einige Methoden zur Reduktion der Codierungskomplexität zeigen. Wir werden uns auch mit der Leistungsgrenze des LDPC beschäftigen und am Ende mit der praktischen Anwendung der LDPC-Decodierung zur Decodierung der Bitebenen empfangener Code-Wörter, wenn die Wahrscheinlichkeitsverteilungsfunktion des Kanalrauschens eine Laplacesche Verteilung aufweist.

4.1 Einleitung

LDPC ist eine Art von linearen Block-Codes und wurde von Robert Gallager [R.G63] in seiner Doktorarbeit 1962 vorgeschlagen. Der Name kommt von der charakteristischen Eigenschaft der dazugehörenden Paritäts-Prüfmatrix *H*, die nur wenige Einsen im Vergleich zur Anzahl der Nullen enthält.

Es ist erwiesen, dass die LDPC-Codes in der Lage sind, sich der Kanalkapazität stark anzunähern. Dies führt zu ihrer Nutzung für Standardanwendungen wie zuletzt DVB-S2 (die zweite Generation der Digitalen Videoübertragung über Satellit). Mackay zeigte, dass LDPC sich der Shannon-Kapazitätsgrenze exponentiell annähern kann, je länger der Code $\boldsymbol{N}$ wird.

Gallager [Gal62] zeigte, dass die minimale Hamming-Distanz zwischen den Code-Wörtern mit längerer Code-Wortlänge *N* steigt, wenn das Gewicht der Spalten und Zeilen (engl: row and column weight) fixiert werden.

Andererseits verhält sich LDPC wie die Zufalls-Codes, die in Shannons Beweis des Kanalcodierungs-Theorems verwendet wurden, das kommt daher, dass LDPC die Kanalkapazität erreicht [Mac99], wenn die Rate des LDPC $\to 0$ geht. Der Nachteil dieser Eigenschaft liegt darin, dass ein sehr langer Code ($N \to \infty$) benötigt wird, um die bestmögliche Leistung zu erzielen. Diese große Blocklänge in Kombination mit der Notwendigkeit einer iterativen Decodierung führt zu einer Latenzzeit, die für die meisten Anwendungen inakzeptabel ist.

LDPC weist eine gute Leistungsfähigkeit auf, die meist über der des Turbo-Codes liegt. Auch die Decodierung ist pro Iterationsschritt weniger kompliziert als beim Turbo-Code. Der wichtige Unterschied zwischen dem LDPC-Code und dem Turbo-Code ist,

dass LDPC melden kann, wenn die Decodierung fehlschlägt, weil ein empfangenes Code-Wort nicht korrekt wiedergegeben werden kann. Der Turbo-Code ist auf einen Grenzwert angewiesen, um die Decodierung zu stoppen, auch nach dem Ende der Decodierung kann der Turbo-Code nicht bestätigen, ob ein Code-Wort korrigiert wurde.

Andererseits ist LDPC-Codierung komplexer als Turbo-Codes und der LDPC-Decoder benötigt mehr Zeit zur Decodierung, da er mehr Iterationsschritte durchführt. Die Komplexität der LDPC-Codierung liegt bei $O(N^2)$, aber bei der Komplexitätsreduktion gibt es Fortschritte (siehe Abschnitt 4.4). Ein Nachteil von LDPC ist auch manchmal das Auftreten von *Error-Floor-Bereichen* durch trapping sets oder Quasi-Code-Wörter (engl: near codeword)(siehe Abschnitt 4.7).

Die letzte Verbesserung von LDPC-Codes war die Einführung nicht-binärer und irregulärer LDPC-Codes.

Die meistgenutzten LDPC sind LDPC-Block-Codes, aber es gibt noch andere Arten von LDPC-Codes wie LDPC-Faltungscodes (engl:LDPC-convolutional codes), die von Felstrom and Zigangirov 1999 [JFZ99] vorgeschlagen wurden. LDPC Faltungscodes weisen auch eine hohe Leistung auf und nutzen Decodierungsalgorithmen, die denen von LDPC-Block-Codes ähnlich sind. Ihr Vorteil Block-Codes gegenüber besteht in ihrer Fähigkeit, mit beliebig langen Informations-Bits zu arbeiten ohne sie in Blöcke zu fragmentieren. Diese Eigenschaft wird genutzt in paketbasierten Kommunikationssystemen, die mit variabler Paketlänge arbeiten, wie IEEE 802.3 Ethernet standards.

4.2 Reguläre Konstruktion eines LDPC-Codes

LDPC-Codes sind definiert im Sinne einer dünn besetzten Paritäts-Prüfmatrix H (engl:sparse matrix), in der die meisten Zahlen Null sind und nur eine kleine Fraktion nicht Null. Da die Matrix dünn besetzt ist, wird nur wenig Speicherplatz zu ihrer Speicherung gebraucht, da zur effizienten Wiedergabe lediglich eine Liste der Nicht-Null-Stellen der Matrix benötigt wird.

Es existieren LDPC-Codes mit zwei unterschiedlichen Strukturen, der regulären und der irregulären. Bei regulär strukturierten LDPC-Codes haben alle Spalten der Prüfmatrix die gleiche Anzahl von Einsen, die Zeilen genauso. Wenn die Anzahl der Einsen in jeder Spalte oder Zeile nicht fest steht, spricht man von einem irregulären LDPC-Code.

In diesem Abschnitt stellen wir eine zufällige Konstruktion für reguläre LDPC-Codes vor. Ein $(\mathrm{N}, \mathrm{j}, \mathrm{K})$ Niedrigdichten-Code ist ein Code mit der Prüfmatrix H, deren Dimension $\mathrm{N} - \mathrm{K} \times \mathrm{K}$ ist, wobei K die Länge der zu codierenden Nachricht ist und N die Code-Wort-Länge, jede Spalte hat dabei die feste Anzahl j von Einsen. Die Länge der Prüfbits für diesen Code beträgt $\mathrm{M} = \mathrm{N} - \mathrm{k}$.

Die Codierungsrate des LDPC ist gegeben durch $\mathrm{R} = \frac{\mathrm{K}}{\mathrm{N}}$. Die durchschnittliche Minimaldistanz steigt linear an mit der Länge des Code-Wortes N. Zur Erstellung einer spezifischen Matrix H für einen regulären Code konstruieren wir zuerst eine zufällige Matrix A zu folgenden Bedingungen:

Wir wissen, dass das Gewicht eines binären Vektors definiert ist als Anzahl der darin enthaltenen Nicht-Null-Elemente. Das Spaltengewicht einer Matrixspalte ist das Gewicht einer Spalte; für das Zeilengewicht gilt das Gleiche. Die Prüfmatrix A kann nach dem Zufallsprinzip generiert werden mit dem geeigneten Spaltengewicht $\mathrm{w_c}$ und den Zeilengewichten $\mathrm{w_r}$, typischerweise ist das Spaltengewicht eine kleine ganze Zahl wie $\mathrm{w_c} = 3$, in [Mac99] zeigte der Autor, dass $\mathrm{w_c} = 2$ ineffektiv ist. Dann versuchen wir, alle Zeilengewichte gleich zu machen, was notwendig ist zur Erfüllung der Gleichung

$$\mathrm{w_c} N = \mathrm{w_r}(N - K) \qquad 4.1$$

Wenn wir die Gleichung 4.1 ausführen und jede Spalte das gleiche $\mathrm{w_c}$ sowie jede Zeile das gleiche $\mathrm{w_r}$ aufweist, beteiligt sich jedes Bit des empfangenen Code-Wortes in $\mathrm{w_c}$ Prüfoperationen und jede Prüfoperation benötigt $\mathrm{w_r}$ Bit. Das führt zu einer guten Decodierung.

Die Prüfmatrix sollte auch keine Zirkel aufweisen, insbesondere keinen Zirkel der Länge 4 im Tanner-Graphen. In anderen Worten sollte die Prüfmatrix nicht mehr als eine Zeile haben, wo in beiden Spalten die Elemente nicht Null sind. Das Vorhandensein von Zirkel führt dazu, dass der Decodierungsalgorithmus nur ungefähre Lösungen berechnen kann, der ungünstigste Zirkel hat dabei die Länge 4. Danach können wir die Gaußsche Elimination mit Spaltenpivotisierung anwenden, um Matrix A umzuwandeln in deren Systemmatrix

$$H_{sys} = [I\ P] \qquad 4.2$$

Wo I die Einheitsmatrix mit der Dimension $(N - K) \times (N - K)$ ist und P eine Matrix mit der Dimension $K \times K$.

Wenn wir A nicht in der systematischen Form H_{sys} konstruieren konnten, bedeutet es, dass unsere Matrix A keinen vollen Rang hat (engl:full rank), in anderen Worten weist sie linear abhängige Zeilen auf. Um dieses Problem zu lösen, müssen wir einen neue Matrix A von Beginn an konstruieren, bis wir Matrix H in der Form der Gleichung 4.2 erhalten.

Die Codierungsrate des regulären Codes ist gegeben durch die Gleichung

$$R = 1 - \frac{\mathrm{w_c}}{\mathrm{w_r}} \qquad 4.3$$

Die Generatormatrix *G* muss dem Encoder angepasst werden. Sobald *A* in der systematischen Form H_{sys} vorliegt, ist es leicht zu bestätigen, dass eine gültige systematische Generatormatrix definiert ist durch

$$G_{sys} = [P^T I] \tag{4.4}$$

Wo *I* die Dimension $K \times K$ hat.

Wir erstellen die Matrix G_{sys} nach Konstruktion der Matrix H_{sys} mittels der Gleichung

$$G_{sys} H_{sys}^T = 0 \tag{4.5}$$

Wir behandelten in diesem Abschnitt die Zufallskonstruktion eines LDPC-Codes, aber es gibt neuere Konstruktionen, bei denen versucht wurde, zusätzliche Strukturen in die Prüfmatrix *H* und/oder die Generatormatrix *G* einzubringen.

4.3 Der Codierungsalgorithmus

Der Code kann betrachtet werden als Abbildung des *K*-Raums in den *N*-Raum mittels der Generatormatrix G. Wenn die zu codierende Nachricht m ist mit der Länge K, ist das Code-Wort gegeben durch die Gleichung

$$c = m.G \tag{4.6}$$

Die Reihen der Matrix G bilden die Basis der Code-Teilmenge (engl:code subspace). Der duale Raum $\mathbb{C}^{\perp}$ besteht aus all denjenigen Vektoren, die senkrecht zu $\mathbb{C}$ stehen; für alle $c \in \mathbb{C}$ und alle $d \in \mathbb{C}^{\perp}$ gilt die Punkt-Multiplikation $< c.d >= 0$. Die Reihen der Matrix *H* bilden eine Basis für $\mathbb{C}^{\perp}$. Folglich gilt für alle $c \in \mathbb{C}$,

$$c.H^T = 0 \tag{4.7}$$

So wird der Code durch die Matrix *G* oder *H* präzisiert, aber keine der Matricen ist einzigartig.

4.4 Fortschritt bei der Komplexitätsreduktion der LDPC-Encoder

Die Sub-Matrix P^T in Gleichung 4.4 ist generell nicht dünn besetzt und die Codierungskomplexität daher recht hoch. Da die G_{sys} Matrix nicht notwendigerweise dünn besetzt ist, wird der Codierungsvorgang die Komplexität $O(N^2)$ aufweisen. Diese

Komplexität steht in Gegensatz zum Fall des Turbo-Codes, dessen Codierungskomplexität sich linear verhält.

Einer der Schwachpunkte der LDPC-Codes ist deren Codierungscomplexität. Viele verschiedene Codierungsschemata wurden für LDPC-Codes zur Komplexitätsreduktion vorgeschlagen [MWD99], [RU01b].

Eins der Codierungsschemata wird in [RU01b] vorgestellt, in diesem Abschnitt diskutieren wir diese Methode. Dieses Schema ermöglicht die Codierung mit einer angemessenen Komplexität, vorausgesetzt, dass der Codierung einige Vorarbeiten vorangehen.

Diese Methode benötigt folgende Schritte:

Die Matrix *H* wird, wie in Abb. 4.1 gezeigt, in neue Form gebracht, was durch Zeilen- und Spalten-Permutation geschieht, was bedeutet, dass die Anzahl der Nicht-Null-Elemente unverändert und daher auch die Minimaldistanz des Codes erhalten bleibt.

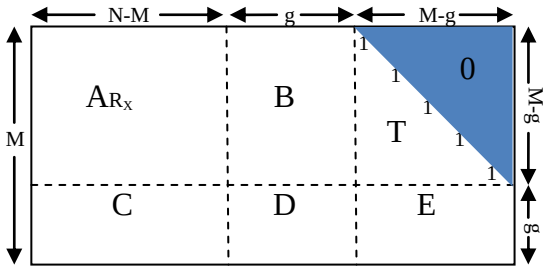


Abb. 4.1: Neue Form der Matrix H zur Konstruktion einer einfachen LDPC-Codierung.

Matrix *T* ist eine niedere diagonale Matrix (engl:lower triangular matrix) mit Einsen entlang der Diagonale. Für dieses Schema gilt, dass je kleiner die Dimension *g* ist, umso niedriger die Komplexität der LDPC-Codierung.

Der nächste Schritt ist die Umwandlung der Matrix *E* in Abb. 4.1 in eine Nullmatrix durch die Multiplikation der Matrix *H* links mit der Matrix $\begin{bmatrix} I & 0 \\ -ET^{-1} & I \end{bmatrix}$. Die neue Form der Matrix ist dann

$$\widehat{H} = \begin{bmatrix} A & B & T \\ -ET^{-1}A + C & -ET^{-1}B + D & 0 \end{bmatrix} \qquad 4.8$$

Die neue Form $\widehat{H}$ ist äquivalent zur alten Form in Abb. 4.1, dies ist darauf zurückzuführen, dass diese Matrixmultiplikation den gleichen Effekt hat, wie die Gaußsche Elimination, daher gehören die Matrices $\widehat{H}$ und *H* zum gleichen LDPC-Code.

Wir nehmen an, dass die Matrix *G* symmetrisch ist, was dazu führt, dass der Code *c* durch $c = [m\ P]$ repräsentiert werden kann, wobei *m* die Nachricht ist, die in LDPC

codiert werden soll, und P die Prüfbits. Die Länge von m ist K Bits und die Länge von P ist $M = N - K$. Teilen wir die Sequenz P in zwei Sequenzen $P = [\mathrm{P}_1\ \mathrm{P}_2]$ auf, können wir den Code c schreiben als

$$c = [m\ \mathrm{P}_1\ \mathrm{P}_2] \tag{4.9}$$

Wir wissen, dass die Codes die Gleichung $c.\widehat{H}^T = \widehat{H}.c^T = 0$ erfüllen sollten, das führt zu den zwei folgenden Gleichungen

$$Am + B\mathrm{P}_1 + \mathrm{TP}_2 = 0 \tag{4.10}$$

$$(-ET^{-1}A + C)m + (-ET^{-1}B + D)\mathrm{P}_1 + 0\mathrm{P}_2 = 0 \tag{4.11}$$

Wenn die Matrix $(-ET^{-1}B + D)$ eine invertierbare Matrix (engl:invertible Matrix) aufweist (in anderen Worten wenn die Matrix nicht singulär ist), können wir die Gleichung 4.11 auflösen, um die Prüfbits der Sequenz P_1zu finden.

$$\mathrm{P}_1 = -(-ET^{-1}B + D)^{-1}\left(-E\left(T^{-1}(Am)\right)\right) + Cm) \tag{4.12}$$

Wir bezeichnen $X = (-ET^{-1}B + D)$ und nehmen an, dass X eine invertierbare Matrix X^{-1} besitzt (also dass X eine nicht-singuläre Matrix ist). Wenn Matrix X singulär ist, sollten wir die Spalten der Matrix $\widehat{H}$ neu ordnen, um eine nichtsinguläre Matrix X zu erhalten. Wir merken auch an, dass der Encoder die Matrix X in einem Arbeitsschritt berechnen und speichern kann. So können wir die Gleichung 4.12 in folgender Form schreiben

$$\mathrm{P}_1 = -X^{-1}\left(-E\left(T^{-1}(Am)\right)\right) + Cm) \tag{4.13}$$

Aus dieser neuen Gleichung sehen wir, dass die Komplexität der Berechnung von P_1 bei $O(g(N-M))$ liegt, da die Dimension der Matrix X gleich $g \times (N-M)$ ist. Das führt dazu, dass wenn g kleiner wird, der Encoder eine kleinere Komplexität zur Berechnung von P_1 braucht und somit auch weniger Speicherplatz zur Speicherung der Matrix X.

Wir können die Gleichung 4.13 in anderer Form schreiben, um ausrechnen zu können, wie hoch die Komplexität der zu ihrer Lösung benötigten Rechnung ist. Die neue Form ist gegeben als

$$T\left[\mathrm{P}_1\left[\frac{1}{-X^{-1}(-E)}\right] - \frac{Cm}{-E}\right] = (Am) \tag{4.14}$$

Die einzige unbekannte Variable in dieser Gleichung ist P_1, das bedeutet die Gleichung 4.14 hat die Form

$$TX_2 = X_3 \qquad 4.15$$

Wobei $X_2 = \left[P_1 \left[\frac{1}{-X^{-1}(-E)}\right] - \frac{Cm}{-E}\right]$ ist und $X_3 = (Am)$. Wir können diese Gleichung durch Rücksubstituion lösen, da die Matrix T, wie bereits erwähnt, eine niedere diagonale Matrix ist, X_3 bekannt und X_2 unbekannt ist. Die Komplexität der Auflösung dieser Gleichung ist gegeben durch $O(N)$.

Nach der Berechnung von X_2 kann P_1 berechnet werden durch die Gleichung

$$P_1 = [-X^{-1}(-E(X_2)) + cm] \qquad 4.16$$

Sobald P_1 bekannt ist, kann P_2 aus der Gleichung 4.10 errechnet werden

$$P_2 = -T^{-1}(Am + BP_1) \qquad 4.17$$

Zur Auflösung der Gleichung schreiben wir sie in der Form

$$TX_4 = X_5 \qquad 4.18$$

Wo $X_4 = P_2$ und $X_5 = (Am + BP_1)$. Die Gleichung kann ebenfalls durch Rücksubstituion gelöst werden, die Komplexität der Lösung ist gegeben mit $O(N)$.

Die Komplexität der LDPC-Codierung mit dieser Methode ist $O(N + g^2)$. Das bedeutet, dass mit dieser Methode die Komplexität von $O(N^2)$ auf $O(N + g^2)$ reduziert wird.

Es existiert eine andere Methode, um die Komplexität der LDPC-Codierung zu verringern, der Name dieser Methode lautet "Niedrigdichten-Generatormatrix"(engl: low density generator matrix =LDGM). Mit dieser Methode erstellen wir die Generatormatrix zunächst in der Form $G = \begin{bmatrix} I \\ P \end{bmatrix}$ und setzen die meisten ihrer Elemente in P gleich Null, so dass eine dünn besetzte Matrix entsteht. Die Prüfmatrix des Codes wird dann die Form $H = [-P \quad I]$ haben und ebenfalls dünn besetzt sein. Das Problem dieser Methode ist, dass die Code-Wörter ein niedriges Gewicht (engl:low weight) aufweisen, was zu error floors führt (Ein error floor ist ein Bereich, in dem die Fehlerwahrscheinlichkeit sich weniger schnell als erwartet der Null annähert). Viele LDPC-Codes, besonders solche, die so konzipiert sind, dass sie nah an der Kanalkapazität ($N \to \infty$ und sehr dünn besetzte Matrix H) arbeiten, zeigen error floors. Aus diesem Grund hat LGDM keine größere Bedeutung, aber der Autor in [GLCF04]

zeigte, dass wenn der Encoder zwei verkettete LGDM verwendet, die Leistung sich verbessert.

4.5 Die Tanner-Graphen

Tanner beschäftigte sich mit LDPC-Codes und mit der Möglichkeit, sie effektiv in einem bipartiten Graphen, dem sogenannten Tanner-Graphen, darzustellen. [Tan81]. Der Tanner-Graph eines LDPC-Codes ist insofern analog zum Gitternetz (engl:trellis) eines Faltungscodes, als dass er eine vollständige Darstellung des Codes bietet und bei der Beschreibung des Decodierungsalgorithmus hilfreich ist.

Ein bipartiter Graph hat zwei Typen von Knoten, variable Knoten und Prüfknoten die an so genannten Kanten (engl:edges) verbunden sind. Die variablen Knoten werden als v-Knoten und die Prüfknoten als c-Knoten (engl: check nodes) bezeichnet, die Verbindungen zwischen ihnen als Kantenverbindungen.

Zum Zeichnen des Tanner-Graphen wenden wir die folgende Regel an:

Der Prüfknoten *j* ist verbunden mit dem variablen Knoten *i*, immer wenn das Element $h_{ji} = 1$ in der Matrix *H* auftritt. Das bedeutet, dass im Graphen $N - K$ Prüfknoten sind, jeder Prüfknoten ist abhängig von einer Prüfgleichung.

Beispiel: Gehen wir aus von der Paritäts-Prüfmatrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Der Tanner-Graph des durch H gegebenen LDPC-Codes ist in Abb. 4.2 dargestellt. Angemerkt sei, dass der Tanner-Graph in diesem Beispiel regulär ist, für die Matrix gilt $w_c = 2$, $w_r = 4$. Daher hat jeder v-Knoten in Abb. 4.2 jeweils 2 Kantenverbindungen und jeder c-Knoten hat 4.

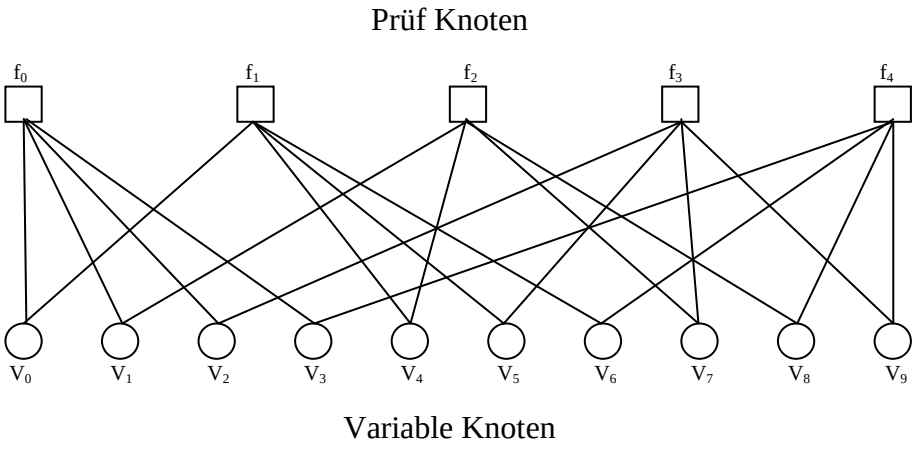


Abb. 4.2: Tanner-Graph für einen Beispiel-Matrix H.

4.6 Weiche LDPC-Decodierung

4.6.1 Das Prinzip des Decodierungsalgorithmus

Die optimale Decodierung (Beispiel: maximum likelihood estimate) des allgemeinen linearen Block-Codes, die nützlich bei der praktischen Anwendung ist, ist bei LDPC-Codes wegen deren hoher Komplexität nicht möglich. Der Niedrigdichten-Aspekt von LDPC-Codes erfordert iterative Decodierung, welche typischerweise eine Leistung nahe an der maximalen likelihood estimate aufweist bei Fehlerraten, die für viele Anwendungen von Interesse sind.

Zur Durchführung der Decodierung nehmen wir an, dass die Menge an Variablen und Prüfknoten ein Baysesches Netzwerk (oder ein Wahrscheinlichkeitsnetzwerk, engl: belief network) bilden. Das Ziel des Decodierungsalgorithmus ist, aus den beobachteten nodes die marginalen Posteriorwahrscheinlichkeiten zu berechnen. Dies wird durchgeführt, indem die posteriori-Wahrscheinlichkeit des Wertes für jeden v-Knoten vorhergesagt wird aus der empfangenen Sequenz $\boldsymbol{r}$ und den Kanaleingeschaften (die Wahrscheinlichkeitsdichtefunktion des kanalfehlers). Die Decodierung ist erzielt, wenn das wiedergegebene Code-Wort die Gleichung 4.7 erfüllt. Wenn der LDPC-Code binär ist, wird die Gleichung 4.7 in modulo 2 sein.

Mathematisch gesehen sucht der Decoder das beste vorausgesagte Code-Wort, welches die Wahrscheinlichkeit $P(\hat{c}|\boldsymbol{r})$ maximiert, gleichzeitig sollte dieses $\hat{\mathrm{c}}$ die Gleichung $\hat{c}.H^T = 0$ erfüllen (in anderen Worten genügt $\hat{c}$ der Paritätsprüfung), wobei $\boldsymbol{r}$ das empfangene Code-Wort mit *n* Bits ist, bezeichnet mit $\mathbf{r} = [\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_\mathrm{n}]$. Die Gleichung $\hat{c}.H^T = 0$ ist erfüllt, wenn alle c-Knoten erfüllt sind, dann kann der

Decodierungsprozess in der Form $P(\hat{c}|\boldsymbol{r}, alle\ Pr\ddot{u}fknoten\ werden\ erf\ddot{u}llt)$ aufgeschrieben werden.

Der Prozess kann gesehen werden als Nachrichtenaustauschalgorithmus (engl. Message Passing algorithm, manchmal auch genannt Belief-Ausbreitungsdecodierer, engl: belief propagation, oder Summenproduktalgorithmus oder iterative Decodierung) am bipartiten Graphen mit zwei Arten Knoten, den variablen Knoten und den Prüfknoten. Die direkte Kantenverbindung zwischen dem v-Knoten und dem c-Knoten zeigt die Beziehung zwischen den Knoten, und dass der Zustand des c-Knoten wird vom v-Knoten, an den er gebunden ist, bestimmt.

4.6.2 Der Decodierungsalgorithmus im Detail

Bevor wir den Decodierungsalgorithmus im Detail erklären, sollten wir einige wichtige Arten von Knoten definieren.

- Die Elemente, die mit jeder Zeile der Matrix H korrespondieren, werden als Tests bezeichnet. Das bedeutet, dass der LDPC-Code $N - K$ Tests aufweist. Die Bitmenge, die für die Prüfknote *m* benötigt wird (i.d.R. die Nicht-Null-Elemente der *m*-ten Zeile von H) ist gegeben durch

$$N_m = \{n: H_{mn} = 1\} \tag{4.19}$$

  So können wir den mit Zeile *m* von H korrespondierenden Test schreiben als

$$z_m = \sum_{n \in N_m} c_n \tag{4.20}$$

  Wo $c = [c_1 c_2 \dots c_n]$ das vorhergesagte Code-Wort am Decoder ist.

- Die mit jeder Spalte der Matrix H korrespondierenden Elemente werden als Variable bezeichnet. Das bedeutet, der LDPC-Code hat *N* Variable, die Menge von Tests, die von einer Variablennummer *n* abhängig sind (i.d.R. die Nicht-Null-Elemente der *n*-ten Spalte von *H*) ist gegeben durch

$$M_n = \{m: H_{mn} = 1\} \tag{4.21}$$

Der Decoder nutzt zwei Arten von Wahrscheinlichkeiten zur Ausführung der Decodierung. Die erste Art ist die Wahrscheinlichkeit, dass der v-Knoten c_n einen bestimmten Wert hat (der Wert ist 0 oder 1, wenn von einem binären Code ausgegangen wird), wenn alle Tests, an denen Bit c_n beteiligt ist, erfüllt sind. Wir bezeichnen diese Wahrscheinlichkeit mit $q_n(0)$,$q_n(1)$, diese Wahrscheinlichkeit können wir beschreiben durch die Gleichung.

$$q_n(x) = P(c_n = x|\boldsymbol{r}, \{z_m = 0, m \in M_n\}), x \in \{0,1\} \quad 4.22$$

In der Literatur wird diese Wahrscheinlichkeit Pseudo-Posteriorwahrscheinlichkeit genannt und üblicherweise am Ende jeder Iteration benutzt, um zu bestimmen, ob Bit *n* des empfangenen Code-Wortes 1 oder 0 ist.

Eine andere Wahrscheinlichkeit der gleichen Art ist gegeben durch

$$q_{mn}(x) = P\big(c_n = x\big|\boldsymbol{r}, \{z_{m'} = 0, m' \in M_{n,m}\}\big)x \in \{0,1\} \quad 4.23$$

Wo $M_{n,m}$ die Menge aller c-Knoten ist, die zur Gruppe M_n gehören, ohne den c-Knoten m.

Die zweite Wahrscheinlichkeitsart bezieht sich auf die c-Knoten, wenn die Werte der Code-Wörter gegeben sind.

$$r_{mn}(x) = P(z_m = 0|c_n = x, \boldsymbol{r}), x \in \{0,1\} \quad 4.24$$

Das bedeutet $r_{mn}(x)$ ist die Wahrscheinlichkeit, dass der c-Knoten m erfüllt ist, wenn der Wert des am c-Knoten z_m, beteiligten Bits in Position *n* gleich *x* ist und zugleich der Empfang des Code-Wortes $\boldsymbol{r}$ beobachtet wurde.

Der Decodierungsalgorithmus berechnet die zwei Wahrscheinlichkeitsarten nur für die Nicht-Null-Elemente der Matrix *H*. Der Decodierungsalgorithmus nutzt auch die Information über die v-Knoten, um die Wahrscheinlichkeiten $r_{mn}(x)$ für die c-Knoten zu berechnen. Unter Zuhilfenahme der Information über die c-Knoten berechnet der Decoder dann die neuen Wahrscheinlichkeiten $q_n(x)$ für die v-Knoten.

Diese neue Wahrscheinlichkeit $q_n(x)$ wird genutzt für eine Update der Wahrscheinlichkeit für die c-Knoten und so fort. Der Update-Prozess wird ausgeführt, bis die Gleichung 4.7 erfüllt ist oder bis eine bestimmte Anzahl Iterationsschritte überstiegen wird. Das Update-Prinzip ist in der Abb. 4.3 gezeigt.

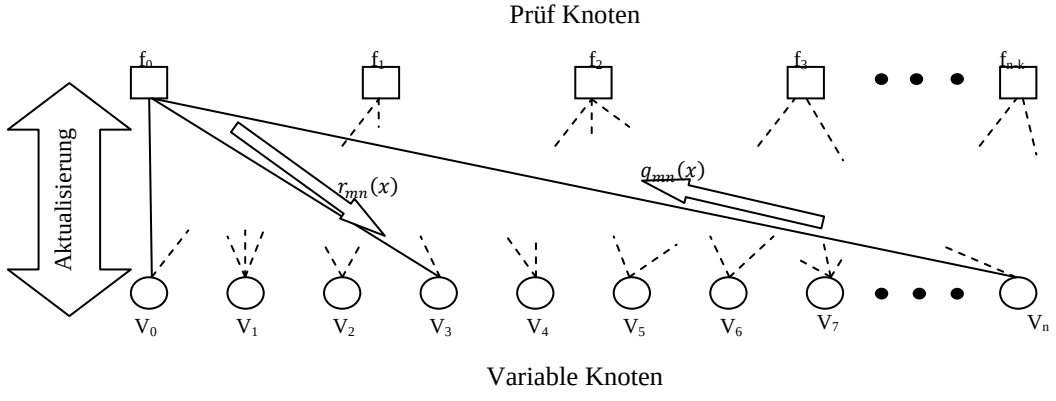


Abb. 4.3: Nachrichtenaktualisierung beim iterativen.

4.6.2.1 Der vertikale Schritt

In Abb. 4.4 sehen wir einen Abschnitt des Tanner-Graphen, in dem die Verbindung eines beliebigen v-Knoten c_n zu dessen c-Knoten gezeigt wird. Dieser variable Knoten c_1 wird dargestellt als Wurzel eines Baums und die anderen Knoten (dargestellt als Bits in Schicht 1), die mit den c-Knoten von c_1 verbunden sind (also die c-knoten z_1, z_2, z_5) als Knoten des Baumes.

Wenn der Tanner-Graph keine Zirkel aufweist, können wir annehmen, dass die Bits in der ersten Schicht (engl:tier) des Baums unabhängig sind. In der Realität treten Zirkel manchmal auf, besonder wenn bei dem Code *N* klein ist im Vergleich zu *K*. wir haben einen Zirkel, wenn ein Bit in Verzweigung 1 mehr als einen Prüfknoten mit der Wurzel des Baums gemeinsam hat.

Als Beispiel in Abb. 4.4 haben wir einen Zirkel der Länge 4, da das Bit c_2 die c-Knoten z_1 und z_5 mit der Baumwurzel c_1 teilt. Wenn *N* sehr groß ist und *K* klein, die Wahrscheinlichkeit für einen Zirkel, insbesondere für einen Zirkel geringer Länge, gering. Der Zirkel zwingt den Decoder in einigen Abschnitten des Graphen lokal zu arbeiten, so dass eine globale Optimallösung unmöglich wird. Die kleinste Zirkellänge in einem gegebenen bipartiten Graphen wird als Taillenweite des Graphen (engl:graph's *girth*) bezeichnet.

Für ein Bit c_n, das am c-Knoten $z_m, m \in M_n$ beteiligt ist, ist im Falle, dass sämtliche c-Knoten unabhängig sind (d.h. wenn der Tanner-Graph keine Zirkel aufweist), ist die Wahrscheinlichkeit, dass der Originalwert von c_n gleich *x* ist, gegeben durch die Gleichung

$$q_n(x) = \beta_n P(c_n = x|\boldsymbol{r}_n) \prod_{m \in M_n} P(z_m = 0|c_n = x, \boldsymbol{r}) \qquad 4.25$$

Wobei x entweder 0 oder 1 ist, wenn der LDPC binär ist.

Die Gleichung 4.25 sagt aus, dass das Originalbit c_n des empfangenen Code-Worts $\boldsymbol{r}$, gegeben bei Kenntnis der Posterior-Kanalwahrscheinlichkeit $P(c_n = x|\boldsymbol{r}_n)$ und durch die Erfüllung aller c-Knoten der Gruppe M_n. β_n eine Konstante ist, damit die Gleichung $\sum_{x'} q_n(x') = 1$ für jedes Bit *n* erfüllt wird, mit der Summe aller möglichen Symbole x' am Ausgang des LDPC-Encoders. Das bedeutet, dass die Gleichung 4.25 in der folgenden Form niedergeschrieben werden kann

$$q_n(x) = \frac{P(c_n = x|\boldsymbol{r}_n) \prod_{m \in M_n} P(z_m = 0|c_n = x, \boldsymbol{r})}{\sum_{x'} P(c_n = x'|\boldsymbol{r}_n) \prod_{m \in M_n} P(z_m = 0|c_n = x', \boldsymbol{r})} \qquad 4.26$$

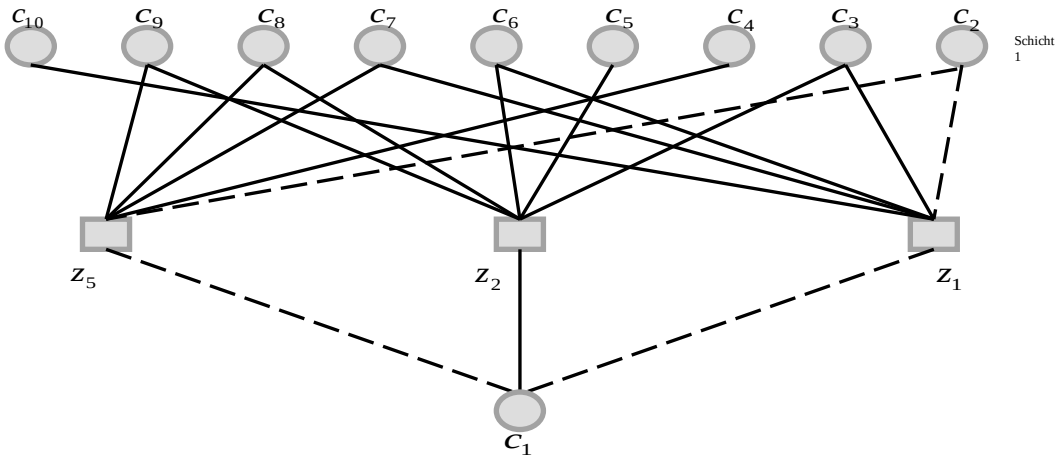


Abb. 4.4: Teilmenge des Tanner-Graphen mit Zirkel (c_1, z_1, c_2, z_5), mit c_1 als Wurzel.

Der Faktor $\prod_{m\in M_n} P(z_m = 0|c_n = x|\boldsymbol{r})$ gibt die Informationsmenge des Bits c_n wieder, die sich auf die durch den Code eingeführte Struktur (eingeführt durch die Struktur der Matrix *H*) bezieht. Der Faktor $P(c_n = x|\boldsymbol{r}_n)$ drückt die Informationsmenge über das Bit c_n aus, die auf dem empfangenen Bit $\boldsymbol{r}_n$ des Code-Worts beruht. $P(c_n = x|\boldsymbol{r}_n)$ wird bezeichnet als Posterior-Kanalwahrscheinlichkeit (engl: posterior channel probability).

Die Wahrscheinlichkeit, dass der *m*-te c-Knoten bei dem gegebenen Bit c_n erfüllt ist, ist gegeben durch die Gleichung

$$r_{mn}(x) = P(z_m = 0|c_n = x, \boldsymbol{r}) \qquad 4.27$$

Wir sehen, dass diese Wahrscheinlichkeit von allen Bits der N_m abhängt, die am c-Knoten z_m beteiligt sind. Bei Einsetzung dieser Gleichung und unter der Annahme, dass sämtliche c-Knoten unabhängig sind (wenn im Tanner-Graphen keine Zirkel auftreten), können wir die Gleichung 4.25 in die Form bringen

$$q_n(x) = \beta_n P(c_n = x|\boldsymbol{r}_n) \prod_{m\in M_n} r_{mn}(x) \qquad 4.28$$

Beim vertikalen Schritt sendet jeder v-Knoten die Wahrscheinlichkeit $q_{mn}(x)$ an die c-Knoten Nummer *m*. $q_{mn}(x)$ ist gegeben durch

$$q_{mn}(x) = \alpha_{mn} P(c_n = x|\boldsymbol{r}_n) \prod\nolimits_{m\in M_{n,m}} r_{mn}(x) \qquad 4.29$$

Wo $M_{n,m}$, wie bereits erwähnt, die Menge der c-Knoten ist, die zu der Gruppe M_n gehören, ohne den c-Knoten *m*, und α_{mn} konstant ist, um $\sum_{x'} q_{mn}(x') = 1$ für jedes Paar (m, n) zu erzielen.

Wir können die Gleichung 4.29 in die Form bringen

$q_{mn}(x) =$
$P(c_n = x|alle\ Knoten\ abhängig\ von\ c_n, außer\ Knote\ z_m, sind\ erfüllt)$

Das bedeutet, $q_{mn}(x)$ gibt die Sicht über den Wert des Bits c_n von anderen c-Knoten wieder.

Die Berechnung der Gleichung 4.29 ergibt, dass $w_c - 1$ Elementen in der Gruppe $M_{n,m}$ enthalten sind (wo w_c das Spaltengewicht der Matrix *H* ist), und somit zur Berechnung von $q_{mn}(x)$ wir w_c Multiplikationen durchführen müssen. Für jedes Nicht-Null-Element der Matrix *H* müssen wir das $q_{mn}(x)$ berechnen, somit müssen wir für den LDPC-Code (N, w_c, K) jeweils $w_c N$ Werte für q_{mn} berechnen. Die Berechnung eines jeden q_{mn} benötigt jeweils $O(w_c)$ Operationen, zur Berechnung aller q_{mn} der Matrix *H* müssen wir somit $O(w_c^2 N)$ Operationen durchführen.

Zum Update von $q_{mn}(x)$ wird die Berechnung des Produkts $r_{mn}(x)$ die *n*-te Spalte der Matrix *H* herunter benötigt, daher wird bei dieesem Update-Vorgang vom vertikalen Schritt gesprochen.

4.6.2.2 Der horizontale Schritt

Wir können die Gleichung 4.27 in neuer Form aufschreiben, nämlich

$$r_{mn}(x) = \sum_{\{x_{n'}, n' \in N_{m,n}\}} P(z_m = 0, \{c_{n'} = x_{n'}, n' \in N_{m,n}\} | c_n = x, \boldsymbol{r}) \qquad 4.30$$

Wo $N_{m,n}$ die gleichen v-Knoten in der Gruppe N_m sind, aber ohne den v-Knoten n. $\{x_{n'}\}$ eine Sequenz ist mit der Länge $|N_n|$-1. Die Summe in der Gleichung 4.30 ist über alle möglichen Sequenzen $\{x_{n'}\}$, das bedeutet, die Summe gilt über alle $q^{|N_n|-1}$ möglichen Werte, wobei q die Anzahl der Symbole am Ausgang des LDPC-Encoders ist, bei einem binären LDPC ist $q = 2$.

Aus der Wahrscheinlichkeitstheorie wissen wir, dass

$$P(A, B|C, D) = \mathrm{P(A|C, D, B)P(B|C, D)} \qquad 4.31$$

Und wenn *B* unabhängig ist von C, lautet die Gleichung

$$P(A, B|C, D) = \mathrm{P(A|C, D, B)P(B|D)} \qquad 4.32$$

In der Gleichung 4.30 ist die Wahrscheinlichkeit der Sequenz $\{c_{n'}\}$ unabhängig von der Wahrscheinlichkeit des empfangenen Bit c_n**.** Daher können wir Gleichung 4.32 nutzen, um die Gleichung 4.30 in die Form zu bringen

$$r_{mn}(x) = \sum_{\{x_{n'}, n' \in N_{m,n}\}} [P(z_m = 0 | c_n = x, \{c_{n'} = x_{n'}, n' \in N_{m,n}\}, \boldsymbol{r}) \times P(\{c_{n'} = x_{n'}, n' \in N_{m,n}\} | \boldsymbol{r})] \quad 4.33$$

Bei der Annahme, dass im Tanner-Graphen kein Zirkel auftritt, sind die Bits der Sequenz $\{x_{n'}\}$ voneinander unabhängig. In dem Fall ist die Wahrscheinlichkeit der Sequenz ein Vielfaches der Wahrscheinlichkeit eines jeden Bits der Sequenz. Und uns ist bekannt, dass die Wahrscheinlichkeit für die Erfüllung des c-Knotens unabhängig ist vom beobachteten Code-Wort $\boldsymbol{r}$ bei gegebenen Werten der Bits c_n, $c_{n'}$, was mathematisch bedeutet, dass

$$P\big(z_m = 0 \big| c_n = x, \{c_{n'} = x_{n'}, n' \in N_{m,n}\}, \boldsymbol{r}\big) = P(z_m = 0 | c_n = x, \{c_{n'} = x_{n'}, n' \in N_{m,n}\}) \quad 4.34$$

Das führt dazu, dass Gleichung 4.33 in folgende Form gebracht wird

$$r_{mn}(x) = \sum_{\{x_{n'}, n' \in N_{m,n}\}} [P(z_m = 0 | c_n = x, \{c_{n'} = x_{n'}, n' \in N_{m,n}\}) \times \prod_{l' \in N_{m,n}} P\,(c_{l'} = x_{l'} | \boldsymbol{r})] \quad 4.35$$

Wir wissen auch, dass die Wahrscheinlichkeit für die Erfüllung eines c-Knotens, wenn alle Bits von dessen Prüfgleichung gegeben sind, entweder bei 1 oder 0 liegt, in Abhängigkeit davon, ob die Bedingung für die Bits in N_m tatsächlich erfüllt wird.

Der c-Knoten *m* ist erfüllt, wenn die Summe der Bits in der Sequenz $\{x_{n'}\}, n' \in N_{m,n}$ dem Wert des Bits c_n entspricht, das heißt in anderen Worten, dass im Falle eines binären LDPC der c-Knoten erfüllt ist, wenn die Summe der Bits in N_m gleich 0 in modulo 2 ist. So ergeben sich zwei Zustände für die Gleichung

$$P\big(z_m = 0 \big| c_n = x, \{c_{n'} = x_{n'}, n' \in N_{m,n}\}\big) = \begin{cases} 1 \;\; wenn \;\; x_n = \sum_{n' \in N_{m,n}} x_{n'} \\ 0 \;\; sonst \end{cases} \quad 4.36$$

In Abhängigkeit von dieser Gleichung können wir die Gleichung 4.35 in der folgenden Form aufschreiben

$$r_{mn}(x) = \sum_{\{x_{n'}, n' \in N_{m,n}\}, x = \sum_l x_l} 1 \times \prod_{l \in N_{m,n}} P\left(c_l = x_l | \boldsymbol{r}\right)] \qquad 4.37$$

Die Summe in dieser Gleichung gilt nicht für alle möglichen Sequenzen $\{x_{n'}\}$, sondern nur für diejenigen, die die Gleichung $x_n = \sum_{n' \in N_{m,n}} x_{n'}$ erfüllen. Somit ist $r_{mn}(x)$ die Gesamtwahrscheinlichkeit (engl:total probability) aller Sequenzen $\{x_{n'}\}, n' \in N_{m,n}$ der Länge $|N_n|$-1 deren Summe der Bit in x (in der alten Gleichung bezeichnet mit dem Symbol x_n) entspricht. Wo die Wahrscheinlichkeit für jedes Bit $x_{n'} = t$ (mit t entweder 0 oder 1 im binären Fall) in jeder Sequenz gegeben ist durch die Posterior-Kanalwahrscheinlichkeit.

$$P(x_{n'} = t) = P(c_{n'} = t | \boldsymbol{r}) \qquad 4.38$$

Um die Gleichung 4.37 zu vereinfachen, diskutieren wir ein neues Parameter wie folgt: wir definieren die Summe der ersten L Bits der Sequenz $\{x_{n'}\}$ durch die Gleichung:

$$\zeta(L) = \sum_{i=1}^{L} x_{N_{m,n}(i)} = \sum_{l \in N_{m,n}} x_l \qquad 4.39$$

Wo der Maximalwert für L die Anzahl der Elemente in $N_{m,n}$ ist.

Wenn der LDPC binär ist, sind für ζ Werte von 0 oder 1 möglich. Ist der LDPC in einem Galois-Feld GF(3), sind Werte für ζ von 0 oder 1 oder 2 möglich.

Wir können die möglichen Werte für ζ in einem Gitternetz (engl:trellis) darstellen. Der Zustand dieses Gitternetzes hängt vom Galois-Feld unseres LDPC ab. In Abb. 4.5 stellen wir ein Beispiel mit 3 Zuständen dar (LDPC in GF(3)).

Wir sind in dieser Abbildung davon ausgegangen, dass das Gitternetz bei $\zeta(0) = 0$ beginnt. Wir sehen auch, dass $\zeta(2)$ drei mögliche Werte annimmt, jeweils in Abhängigkeit von den Werten von $\zeta(1)$ und $x_{N_{m,n}(2)}$. Den Endzustand des Gitternetzes stellt $\zeta(L)$ dar.

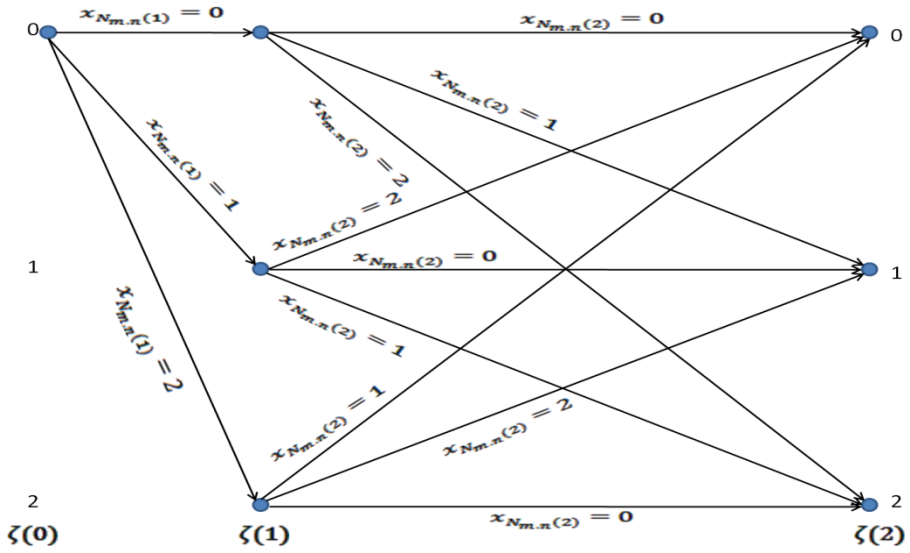


Abb. 4.5: Das Trellis-Diagrammfür die Ermittlung von $r_{mn}(x)$ in GF(3).

Durch Nutzung des Gitternetzes und die Feststellung, dass sich die Summe in Gleichung 4.39 im Index der Summe aus Gleichung 4.37 wiederfindet, konnte festgestellt werden, dass $r_{mn}(x)$ die Wahrscheinlichkeit ist, dass $x = \zeta(L)$, wobei die Wahrscheinlichkeit über jeden Pfad durch das Gitternetz berechnet wurde. Als Beispiel sehen in Abb. 4.5, dass

$$\begin{aligned} r_{mn}(x=0) &= P(\zeta(L)=0) = P(\zeta(2)=0) \\ &= P(\{x_{n'}\}=[0\ 0]) + P(\{x_{n'}\}=[1\ 2]) + +P(\{x_{n'}\}=[2\ 1]) \end{aligned} \tag{4.40}$$

Das bedeutet, dass das Ereignis $\zeta(2) = 0$ über drei Pfade stattfinden kann. Wir können die letzte Gleichung in folgender Form niederschreiben

$$\begin{aligned} r_{mn}(x=0) = P\big(\zeta(1)=0, x_{N_{m,n}(2)}=0\big) + P\big(\zeta(1)=1, x_{N_{m,n}(2)}=2\big) \\ + P\big(\zeta(1)=2, x_{N_{m,n}(2)}=1\big) \end{aligned} \tag{4.41}$$

Bei Annahme, dass die Bits der Sequenz $\{x_{n'}\}$ voneinander unabhängig sind, können wir die letzte Gleichung überführen in die Form

$$\begin{aligned} r_{mn}(x=0) = \quad & P(\zeta(1)=0)P\big(x_{N_{m,n}(2)}=0\big) \\ & + P\big(\zeta(1)=1\big)P(x_{N_{m,n}(2)}=2\big) \\ & + P(\zeta(1)=2)P(x_{N_{m,n}(2)}=1) \end{aligned} \tag{4.42}$$

Gehen wir aus von der Wahrscheinlichkeit

$$w_k(x) = P(\zeta(k)=x) \tag{4.43}$$

Wo k gleich ist oder kleiner als $L = |N_n| - 1$.

Bei Nutzung der Gleichung 4.43 können wir die Gleichung 4.42 umschreiben als

$$\begin{aligned} r_{mn}(x=0) = w_L(0) = w_{L-1}(0)P(x_{N_{m,n}(L)} = 0) + \\ w_{L-1}(1)P(x_{N_{m,n}(L)} = 2) + w_{L-1}(2)P(x_{N_{m,n}(L)} = 1) \end{aligned} \quad 4.44$$

Aus dieser Gleichung können wir schließen, dass

$$r_{mn}(x) = w_L(x) \quad 4.45$$

Um die Wahrscheinlichkeit in Gleichung 4.42 auszurechnen, brauchen wir die Wahrscheinlichkeit der Bits der Sequenzen, gegeben durch $P(x_{n'}) = P(c_{n'}|r_{n'})$, aber die Bits der Sequenz $\{x_{n'}\}$ können nicht nur mit dem c-knoten *m* verbunden sein, sondern auch mit anderen c-knoten. Der Decodierungsalgorithmus basiert darauf, dass von jedem v-Knoten n' zum c-knoten *m* Information gesendet wird über die Wahrscheinlichkeit, dass sich dieser v-Knoten im Zustand *x* befindet, wenn die Sicht aller c-Knoten der Gruppe $N_{m,n}$ gegeben ist (also ohne die Sicht des c-Knoten *m*). So werden wir im Decodierungsalgorithmus die folgende Gleichung nutzen

$$q_{mn'}(x) = P(c_{n'} = x|\boldsymbol{r}) \quad 4.46$$

Unter dieser Annahme können wir die Gleichung 4.44 in die Form überführen

$$\begin{aligned} r_{mn}(x=0) = w_L(0) = w_{L-1}(0)q_{m,N_{m,n}(L)}(0) + \\ w_{L-1}(1)q_{m,N_{m,n}(L)}(2) + w_{L-1}(2)q_{m,N_{m,n}(L)}(1) \end{aligned} \quad 4.47$$

Bei einem binären LDPC lautet diese Gleichung

$$\begin{aligned} r_{mn}(0) = r_{mn}(x=0) = w_L(0) = w_{L-1}(0)q_{m,N_{m,n}(L)}(0) + \\ w_{L-1}(1)q_{m,N_{m,n}(L)}(1) \\ r_{mn}(1) = r_{mn}(x=1) = w_L(0) = w_{L-1}(1)q_{m,N_{m,n}(L)}(0) + \\ w_{L-1}(0)q_{m,N_{m,n}(L)}(1) \end{aligned} \quad 4.48$$

Durch das Ausrechnen der Differenz von $r_{mn}(0)$ und $r_{mn}(1)$ erhalten wir

$$\begin{aligned} r_{mn}(0) - r_{mn}(1) \\ = (w_{L-1}(0) - w_{L-1}(1))(q_{m,N_{m,n}(L)}(0) - q_{m,N_{m,n}(L)}(1) \end{aligned} \quad 4.49$$

Wir können jedoch $w_{L-1}(0)$ und $w_{L-1}(1)$ in Abhängigkeit von Gleichung 4.48 ausrechnen, indem wir den Parameter $L-1$ durch L ersetzen, dies führt dazu, dass wir die letzte Gleichung in folgender Form schreiben

$$r_{mn}(0) - r_{mn}(1) = \prod_{i=1}^{k} (q_{m,N_{m,n}(i)}(0) - q_{m,N_{m,n}(i)}(1)) \qquad 4.50$$

Führen wir die Differenzwahrscheinlichkeiten ein

$$\delta r_{mn} = r_{mn}(0) - r_{mn}(1) \qquad 4.51$$

$$\delta q_{ml} = q_{ml}(0) - q_{ml}(1) \qquad 4.52$$

Bei Einsetzen dieser Differenzwahrscheinlichkeiten können wir die Gleichung 4.50 in die Form überführen

$$\delta r_{mn} = \prod_{l' \in N_{m,n}} \delta q_{m,l'} \qquad 4.53$$

Durch Einbeziehung der Tatsache dass $r_{mn}(0) + r_{mn}(1) = 1$ und der Gleichungen 4.51, 4.52 stellen wir fest, dass

$$r_{mn}(0) = \frac{1 + \delta r_{mn}}{2} \qquad 4.54$$

$$r_{mn}(1) = \frac{1 - \delta r_{mn}}{2} \qquad 4.55$$

Die Prüfknoten nehmen die Werte $q_{ml}(0), q_{ml}(1)$ von den variablen Knoten im vertikalen Schritt an und können anschließen im horizontalen Schritt $\delta r_{mn}, r_{mn}(0), r_{mn}(1)$ durch Einsetzung in die Gleichungen 4.53, 4.54, 4.55 berechnen. Die c-Knoten senden die Werte von $r_{mn}(0), r_{mn}(1)$ an die v-Knoten in jedem horizontalen Schritt.

Dieser Schritt wird als horizontaler Schritt bezeichnet, da in diesem Schritt für jedes Nicht-Null-Element (m, n) der Matrix *H* das Produkt der Gleichung 4.53 über die ganze Zeile Nummer *m* berechnet werden muss. Zur Aktualisierung von δr_{mn} in der Matrix *H*, müssen wir $(w_r M)$ Elemente aktualisieren. Die Aktualisierung eines jeden Elements benötigt $O(w_r - 1)$ Operationen.

4.6.3 Die Zusammenfassung des Decodierungsalgorithmus für einen binären LDPC

Der Decoder-Input besteht aus: die maximale Iteration *L*, die Kanal-Posteriorwahrscheinlichkeit $P(c_n = x|r_n)$, das empfangene Code-Wort $\boldsymbol{r}$, die Paritäts-Prüfmatrix *H*.

Beispielsweise, wenn der Rauschkanal ein BSC ist mit gleichverteilten binären Zufallsvariablen-Inputs und die Änderungswahrscheinlichkeit des BSC bei 0.5 liegt, ist die Kanal-Posteriorwahrscheinlichkeit $P(c_n = 0|r_n) = P(c_n = 1|r_n) = 0.5$.

Die Decodierungsschritte:

1. **Initialisierung:**
   Für jedes Nicht-Null-Element (*m*,*n*) in Matrix *H* gilt $q_{mn}(x) = P(c_n = x|r_n)$. Die Wahrscheinlichkeit der Bits bei gegebenen Prüfknoten, gleichgesetzt mit der Kanal-Posteriorwahrscheinlichkeit.
2. **Horizontaler Schritt:**
   Für jedes Nicht-Null-Element (*m*,*n*) in Matrix *H* berechne man $\delta q_{mn} = q_{mn}(0) - q_{mn}(1)$, dann berechne man $\delta r_{mn} = \prod_{l' \in N_{m,n}} \delta q_{m,l'}$, und dann $r_{mn}(0) = \frac{1+\delta r_{mn}}{2}$, $r_{mn}(1) = \frac{1-\delta r_{mn}}{2}$.
3. **Vertikaler Schritt:**
   Für jedes Nicht-Null-Element (*m*,*n*) in Matrix *H*, berechne man $q_{mn}(x = 0) = q_{mn}(0) = \alpha_{mn} P(c_n = 0|\boldsymbol{r}_n) \prod_{m' \in M_{n,m}} r_{m'n}(0)$ und $q_{mn}(x = 1) = q_{mn}(1) = \alpha_{mn} P(c_n = 1|\boldsymbol{r}_n) \prod_{m' \in M_{n,m}} r_{m'n}(1)$.
   wobei α_{mn} so gewählt ist, dass $q_{mn}(0) + q_{mn}(1) = 1$. Dann berechne man die Wahrscheinlichkeit $q_n(0), q_n(x)$ (wir bezeichnen sie als Pseudoposteriorwahrscheinlichkeit) aus

   $q_n(x = 0) = q_n(0) = \beta_n P(c_n = 0|\boldsymbol{r}_n) \prod_{m' \in M_n} r_{m'n}(0)$ und $q_n(1) = \beta_n P(c_n = 1|\boldsymbol{r}_n) \prod_{m' \in M_n} r_{m'n}(1)$. Wobei β_n so gewählt wird, dass $q_n(0) + q_n(1) = 1$.
4. **Testschritt:**
   Man setze fest, dass das für vorhergesagte Bit *n* gilt $\hat{c}_n = 0$ falls $q_n(0) > 0.5$ ist, und $\hat{c}_n = 1$ falls $q_n(0) \leq 0.5$. Wenn $H.\hat{\boldsymbol{c}} = 0$ ist, stoppe man und setze fest, dass das Original-Code-Wort die Sequenz $\hat{\boldsymbol{c}}$ sei. Falls $H.\hat{\boldsymbol{c}} \neq 0$ und die Anzahl der Iterationen kleiner ist als *L*, gehe man zum horizontalen Schritt. Falls $H.\hat{\boldsymbol{c}} \neq 0$ und die Anzahl der Iterationen gleich *L*, stoppe man und melde einen Decodierungsfehler.

Die Abb. 4.6 zeigt die Decodierungsleistung von LDPC-Codes im binären AWGN Kanal mit dem Gaußschen Rauschen $\sim\mathcal{N}(0,\frac{N_0}{2})$, und dem Energiebetrag E_b, der pro übertragenes Bit nötig ist.

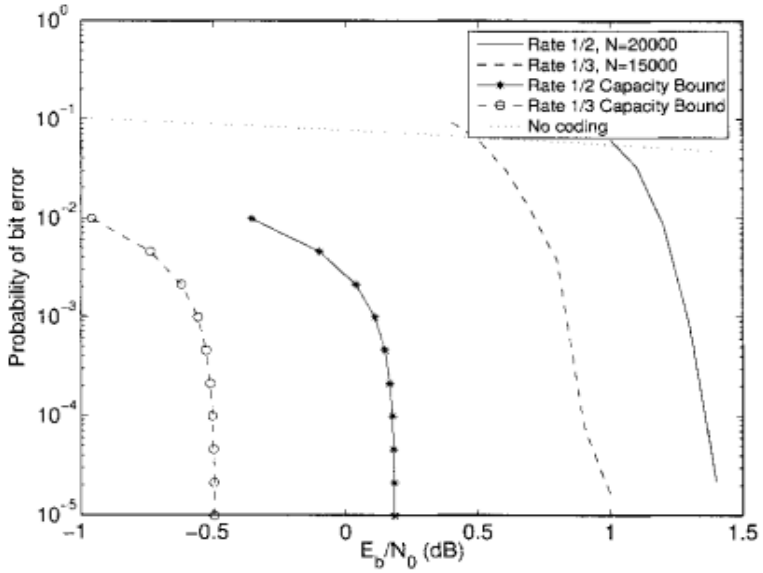


Abb. 4.6: Decodierungsleistung für LDPC-Codes mit den Raten R=1/2 und R=1/3.Quelle: [Moo05].

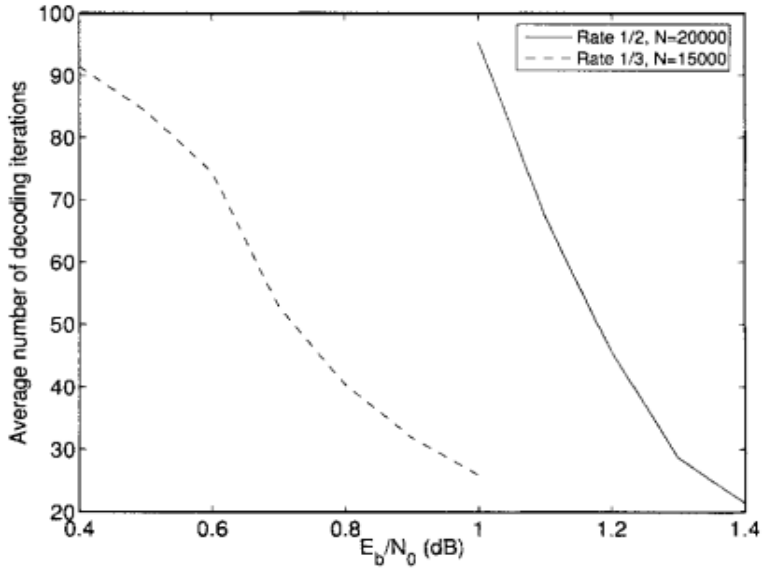


Abb. 4.7: Durchschnittliche Anzahl von Decodierungsiterationen für LDPC mit R=1/3 N=15000 und R=1/2 N=20000. Quelle: [Moo05].

Wir sehen in Abb. 4.7 die durchschnittliche Anzahl von Iterationen für die komplette Decodierung. Wir sehen, dass bei einer kleinen Codierungsrate des LDPC (das bedeutet, wenn N wesentlich größer ist als K) der Decoder eine kleinere Anzahl Iterationen braucht um bei gleichem E_b/N_0 eine verlustfreie Decodierung auszuführen. In diesem tritt aber das Problem auf, dass jede Iteration länger dauert, was für viele Anwendungen inakzeptabel ist.

4.7 Der error-floor des LDPC

Die Fehlerraten-Kurven iterativ decodierter Codes weisen allgemein einen Bereich auf, wo die Steilheit der Kurve absinkt, während der Kanal-SNR ansteigt (das heist wenn die Änderungswahrscheinlichkeit ϵ des BSC abfällt).

Für einige Anwendungen stellt der error-floor kein Problem dar. Für Anwendungen, die extrem niedrige Fehlerraten benötigen (z. B. magnetische oder optische Speicherung), ist ein error-floor inakzeptabel.

Bei den meisten Kanalcodierungen kann diese Abflachung gelegentlich mit einer kleinen Minimal Hamming-distanz in Verbundung gebracht werden, aber beim LDPC ist es möglich, dass eine Abflachung in der Fehlerraten-Kurve auch bei großer Minimal Hamming-distanz entsteht. Normalerweise leidet ein ML-Decoder nicht unter error-floors, aber für iterative Decoder wie Turbo und LDPC stellen die error-floors ein Problem dar, da diese Decoder lokal mit verteilten Prozessen arbeiten im Gegensatz zum ML-Decoder.

In Abb. 4.8 sehen wir ein Beispiel für einen error-floor bei Nutzung eines binären AWGN, HW steht in der Abbildung für “Hardware”, da die Simulation an einem FPGA-basierten Simulator durchgeführt wurde (siehe [Zha07]) und SPA ist der Summenprodukt-Algorithmus. Der Kurvenbereich vor der Abflachung (engl: slope transition region) wird als Wasserfall-Bereich (engl: waterfall region) der Fehlerraten-Kurve bezeichnet und der Bereich mit der reduzierten Steilheit als Fehlerraten-Floor-Bereich oder Floor-Bereich (engl: error-rate floor region oder floor region).

Wir beobachten auch in , dass obwohl der SPA-Decoder im Wasserfall-Bereich eine um etwa 0.3 dB bessere Leistung zeigt, ein Fehlerraten-Floor vorliegt. Dieser Floor ist den trapping sets des SPA-Decoder zuzurechnen, aber bei Nutzung des MS-Decodierungsalgorithmus tritt er nicht auf. Normalerweise weisen Codes sehr niedrige error-floors auf, die tendenziell etwa ein halbes dB schlechter sind als die Decodierungsgrenze.

Im Fall der Floors bei LDPC-Codes mit großen Minimaldistanzen kann der Floor auf so genannte trapping sets zurückgeführt werden (wie normalerweise im Fall des seriellen Turbo-Codes). Ein (w, v) trapping set ist eine Menge von w variablen Knoten, welche einen Sub-Graphen mit v ungeraden checks (checks haben ungerade Pfade) induziert, so dass im Falle, dass alle w Bits falsch sind, v fehlgeschlagene Paritätsprüfungen auftreten. Das bedeutet, ein trapping set ist eine Ansammlung von Code-Symbolen (variablen Knoten), die nicht korrekt konvergiert.

Das trapping set ist abhängig von der Paritäts-Prüfmatrix des Decoders und von Decoder-Algorithmus. Praktisch identifizieren wir das trapping set unter Simulationsbedingungen folgendermaßen: Eine große, festgellegte Anzahl von

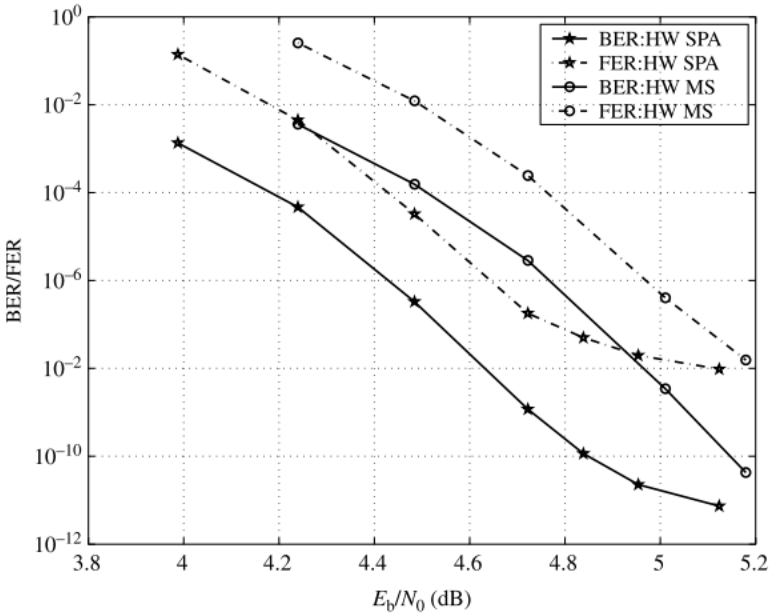


Abb. 4.8: Leistung der Bitfehlerrate und Frame-Fehlerrate eines 0.9(4550, 4096) quasi-zirkularen IRA-Codes mit SPA und min-sum-Decodierung. Quelle: [WER09].

Iterationsschritten (sagen wir 200) wird durchgeführt, außer wenn der Decoder vorher ein Code-Wort findet.

Wenn nach der festgelegten Anzahl Iterationen immer noch keine Konvergenz stattgefunden hat, werden einige zusätzliche Iterationen durchgeführt (sagen wir 30) und das trapping set als die Gesamtmenge aller Bits identifiziert, die während dieser 30 Iterationen nicht korrekt korrigiert werden konnten.

In Abb. 4.9 findet man ein Beispiel für ein trapping set. Normalerweise weisen stark gebundene LDPC (irreguläre LDPC) viele trapping sets auf, was zu schlechten Error-Floor-Leistungen führt. LDPC mit kurzer Blocklänge, hoher Codierungsrate und niedriger Decodierungsgrenze sind für trapping sets teilweise besonders empfindlich.

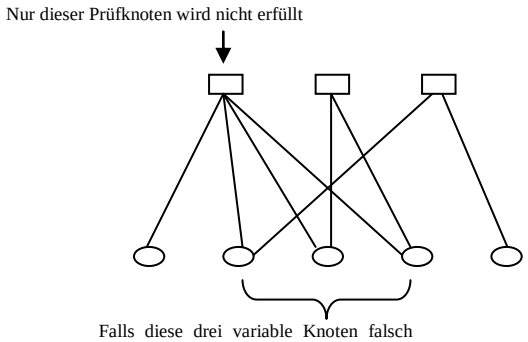


Abb. 4.9: Beispiel für ein trapping set (3,1).

Das Beinahe-Code-Wort (w, v) ist ein Fehlermuster der Länge n mit dem Gewicht w, was zu v fehlgeschlagenen checks führt, das bedeutet, das Syndromgewicht des Fehlermusters ist v. Die Werte von w, v im Beinahe-Code-Wort sind normalerweise klein. Wenn der Wert v klein ist, bedeutet das, dass wenige Prüfgleichungen vom Fehlermuster betroffen sind, in diesem Fall kann der iterative Decoder diese Situation lösen. Wenn der Wert w klein ist, führt es dazu, dass dieses Fehlermuster mehr dazu neigt (das trifft zu, wenn wir annehmen, dass der Rauschfehler im Kanal nicht sehr groß ist).

Der SPA-Algorithmus hat Fehlersituationen, wenn ein Beinahe-Code-Wort auftritt. In diesem Fall könnte der SPA-Algorithmus aus der Situation nicht herauskommen. Einige Bemühungen, den Error-Floor niedriger zu halten, wurden gemacht, indem die Taillenweite des Tanner-Graphen vergrößert wurde, aber der Erfolg war begrenzt. Es gibt ein Kompromiss zwischen Decodierungsgrenzwerten (bei der *Density evolution*-Analysis, sieh Abschnitt 4.11) und bei dem Error-floor-Analysis.

Decodierungstechniken zur Erniedrigung von Floors, besonders für diesen Code, wurden in [Zha07], [HR09] betrachtet.

4.8 Harte LDPC-Decodierung

Der Wahrscheinlichkeitsausbreitungsalgorithmus (engl:belief propagation algorithm) ist der beste Algorithmus unter allen Nachrichtenaustausch-Decodern (engl:message passing decoders). Trotzdem, für praktische Anwendungen auf anderen Kanälen als dem BEC (engl:binary erasure channel), ist der Wahrscheinlichkeitsausbreitungs-algorithmus eher kompliziert und führt oft zu einer Abnahme der Decodergeschwindigkeit. Daher wird oft die harte Decodierung (engl:Hard Decision Decoding) genutzt.

Harte Decodierung wird ausgeführt, wenn die ausgetauschten Nachrichten binär sind im Gegensatz zur weichen Decodierung (engl:soft decision decoder), die mit einer größeren Auswahl an Werten arbeitet.

Wir werden drei Algorithen für harte Decodierungen im BSC beschreiben, alle nach Gallager [R.G63]. Diese Algorithmen wurden genannt Gallager Algorithmus A, Gallager Algorithmus B, und Bit-Flipping-Algorithmus. Kschischang and Aradakani [Ard04] haben gezeigt, dass Algorithmus B optimal ist für reguläre LDPC-Codes und auch optimal für irreguläre LDPC-Codes unter sehr lockeren Bedingungen

Gallagers Algorithmus A:

Der Algorithmus wird in folgenden Schritten durchgeführt:

1.Initialisierung

in diesem Schritt senden alle variablen Knoten n eine Nachricht an ihre Prüfknoten m,

die das für sie als korrekt angenommene Bit enthält. Bei diesem Schritt ist die einzige Information, die der variable Knoten n hat, das korrespondierende empfangene *n-te* Bit der Sequenz $\boldsymbol{r}$.

Die gesendete Information $q_{mn}(x)$ von der Bits zu gegebenen Prüfknoten gleichgesetzt mit dem empfangenen Bit r_n.

$$q_{mn}(x = r_n) = \mathrm{P}(x = r_n) = 1, q_{mn}(x \neq r_n) = \mathrm{P}(x \neq r_n) = 0 \qquad 4.56$$

Wo *m* die Nummer des PrüfKnoten ist und *n* die Nummer des variablen Knotens.

Im Algorithmus der harten Decodierung brauchen die v-Knoten keine Wahrscheinlichkeitswerte q_{mn} zu senden, stattdessen senden sie nur den Wert des empfangenen Bits r_n, stellen wir diese Nachricht dar als $Q_{i,mn}$. Im Initialisationsschritt wird der v-Knoten die gleiche Nachricht an all seine Prüfknoten senden, so dass wir $Q_{i,mn}$ gegen $Q_{i,n}$ austauschen können. So haben wir im Initialisationsschritt

$$Q_{0,n} = r_n \qquad 4.57$$

Wobei die Null in diesem Falle für die Iterationsanummer Null steht. Man merke, dass $Q_{0,n}$ eine binäre Quantität ist.

2.Horizontaler Schritt:

Im zweiten Schritt berechnet jeder c-Knoten m eine Antwort (diese Anwort sollte die Paritäts-Prüfungsgleichung erfüllen) an jeden verbundenen v-Knoten. Die Antwortnachricht enthält das Bit, von dem m annimmt, dass es für diesen v-Knoten n richtig sein unter der Annahme, dass die anderen v-Knoten, die mit dem c-Knoten m verbunden sind, korrekt sind. Diese Annahme hängt von der Addition (mod 2) aller eintreffenden Nachrichten aller anderen v-Knoten außer n.

$$\delta_{i,mn} = \left(\sum_{n \in N_{m,n}} Q_{i,m'n} \right) (mod 2), wo\ m' \in\ N_{m,n} \qquad 4.58$$

Hier ist i wieder die Nummer der Iteration. Man merke, dass das $\delta_{i,mn}$ eine binäre Quantität ist. Führen wir die Wahrscheinlichkeit $p_{i,n}$ ein, bei der es sich um die Wahrscheinlichkeit handelt, dass eine von einem v-Knoten n gesendete Nachricht an den c-Knoten *m* in Gruppe $N_{m,n}$ bei Iteration i gleich 1 ist.

$$p_{i,n} = P(Q_{i,mn} = 1) \qquad 4.59$$

In Abhängigkeit von dieser Wahrscheinlichkeit können wir die folgenden Wahrscheinlichkeiten berechnen

$$r_{i,nm}(1) = P(\delta_{i,mn} = 1) = \frac{1 - [1 - 2p_{i,n}]^{|N_{m,n}|}}{2}$$
$$r_{i,nm}(0) = P(\delta_{i,mn} = 0) = \frac{1 + [1 - 2p_{i,n}]^{|N_{m,n}|}}{2} \quad 4.60$$

Wo $|N_{m,n}|$ die Anzahl der Bits in der Gruppe $N_{m,n}$ ist. Jeder c-Knoten *m* sendet die Werte $\delta_{i,mn}$ an den v-Knoten *n* in Gruppe $M_{n,m}$. Man merke, dass die Gleichung 4.58 in mod2 sicherstellt, dass die Summe der eintreffenden Nachrichten $\{Q_{i,m'n}, m' \in N_{m,n}\}$ und der ausgehenden Nachricht $Q_{i,mn} \rightarrow m$ gleich Null ist.

3.Testschritt:

Wenn alle Testgleichungen erfüllt sind, stoppt der Decodierungsalgorithmus und die Wiederherstellungsbits haben bei der letzten Iteration Nummer i die Werte $Q_{i,n}$. Die Wahrscheinlichkeit, dass alle Prüfgleichungen für einen bestimmten v-Knoten *n* in der Iteration Nummer i erfüllt sind, ist gegeben durch

$$P(alle\ \delta_{i,mn} = Q_{i,n}) = \left[P(Q_{i,mn} = 0)\prod_m r_{i,nm}(0)\right] + \left[P(Q_{i,mn} = 1)\prod_m r_{i,nm}(1)\right] \quad 4.61$$

Wenn nicht alle Prüfgleichungen für alle v-Knoten erfüllt sind, gehen wir zum nächsten Schritt über.

3.Vertikaler Schritt:

Die v-Knoten empfangen die Nachrichten von den c-Knoten und nutzen diese zusätzliche Information um zu entscheiden, ob ihre jeweiligen angenommenen Bitwerte richtig sind. Ein einfacher Weg dazu ist eine Mehrheitsabstimmung.

Nehmen wir an, dass der v-Knoten *n* mit j c-Knoten versehen ist, und für jede Iteration i ein Grenzwert $b_{i,j}$ besteht (der berechnet werden soll), so dass bei einer Iteration i für jeden v-Knoten n mindestens $b_{i,j}$ c-Knoten von n bis auf m die gleiche Information in der vorherigen Iteration senden, dann sendet n diese Information an m; andernfalls sendet n das logische Gegenteil der Information an m.

$$Q_{i,mn} = Q_{i,n} = \begin{cases} d \; if \; \left|\delta_{i-1,m'n} = d\right| \geq \mathrm{b_{i,j}}, \text{wo d} \in \{0,1\} \text{ und } m' \in \mathrm{N_{mn}} \\ d^{\mathrm{c}} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{falls anders} \end{cases} \quad 4.62$$

$\left|\delta_{i-1,m'n} = d\right|$ ist die Anzahl der Nachrichten d von den c-Knoten in der Gruppe N_{mn}, d^{c} ist das logische Gegenteil von d. Wir sehen, dass der v-Knoten die gleiche Nachricht an all seine c-knoten sendet, also können wir $Q_{i,mn}$ gegen $Q_{i,n}$ austauschen. Im Gallager Algorithmus A ist $\mathrm{b_{i,j}}$ gegeben durch

$$\mathrm{b_{i,j}} = 1 \quad 4.63$$

Das bedeutet, dass die ausgehende Nachricht $Q_{i,mn} \rightarrow m$ gleich d gesetzt wird außer wenn alle eintreffende extrinsische Nachrichten $\delta_{i-1,m'n} \rightarrow n$ Widerspruchs ($m' \neq m$), in diesem Falll $Q_{i,mn} \rightarrow m$ mit d^{c} gleichgesetzt wird. Nach dem vertikalen Schritt gehen wir zum horizontalen Schritt über.

Gallager Algorithmus B:

Gallagers Decoderuingsalgorithmus B ist eine Verfeinerung von Algorithmus A, in dem die ausgehenden Nachrichten der variable Knoten $Q_{i,mn}$ nicht benötigen, dass alle extrinsischen Nachrichten $\delta_{i-1,m'n}$ dem Wert d widersprechen, um gleich d^{c} zu sein. Wenn mindestens $\mathrm{b_{i,j}}$ solche Nachrichten vorhanden sind, wird $Q_{i,mn}$ gleich d^{c} gesetzt, andernfalls setzt man es gleich d. Die Update-Regel für v-Knoten ist somit durch Gleichung 4.62 gegeben ohne die Gleichung 4.63.

Gallager zeigte, dass bei regulären LDPC-Codes (d_v, d_c) der optimale Wert für $\mathrm{b_{i,j}}$ die kleinste ganze Zahl b ist, die Gleichung 4.64 erfüllt

$$\frac{1-\varepsilon}{\varepsilon} \leq \left[\frac{1+(1-2p)^{d_c-1}}{1-(1-2p)^{d_c-1}}\right]^{2b-d_c+1} \quad 4.64$$

Wo ε hängt von der Änderungswahrscheinlichkeit des BSC, und p ist die Fehlerwahrscheinlichkeit nach Verarbeitung eines Digits im i-ten schicht (engl:tier).

Aus dieser Gleichung sehen wir, dass der beste Wert für $\mathrm{b_{i,j}}$ von der ε und p abhängt.

Der bit-flipping-Algorithmus für den BSC:

Dieser Algorithmus hängt ab von „flips“ (Umkehrungen) von Bits im empfangenen Code-Wort, die in mehr als eine bestimmten Anzahl fehlgeschlagener Paritätsprüfungen beteiligt waren. Der erste Schritt im Bit-Flipping-Algorithmus ist die Abschätzung aller Paritätsprüfungsgleichungen in der Prüfmatrix H und der zweite Schritt ist die Flipping-Regel.

Der Flipping-Schritt wird dann mit dem modifizierten empfangenen Wort wiederholt, bis alle Paritätsprüfungen erfüllt sind oder bis eine Maximalanzahl von Iterationen ausgeführt worden ist.

Durch die Elemente des Syndroms, $s = rH^T$ kennzeichnen wir die fehlgeschlagenen Paritätsprüfungsgleichungen. Die Nummer der fehlgeschlagenen Paritätsprüfungen für jedes Code-Bit enthalt im Vektor $f = sH$, wo der Vektor f die Länge n ist. Die Operationen zur Berechnung des Vektors sind über den ganzen Zahlen und nicht in den binären Operationen.

Wir können die Schritte des Bit-Flipping-Algorithmus wie folgt zusammenfassen

Schritt1:
Man berechne das Syndrom $s = rH^T$ über den binären Raum GF(2). Falls $s = 0$, gibt es keine fehlgeschlagenen Paritätsprüfungen, kein Bit muss umgekehrt werden, der Algorithmus soll stoppen und das richtige Code-Wort ist r.

Schritt2:
berechne den Vektor $f = sH$ über dem ganzzahligen Raum (engl:integer space) $\mathbb{Z}$.

Schritt3:
identifiziere Elemente von f, die größer sind als ein vorbestimmter Grenzwert und kehre diejenigen Bits von r um, die mit diesen Elementen korrespondieren

Schritt4:
wir gehen zu Schritt 1 zurück und nutzen das aktualisierte r, wenn wir die maximale Anzahl von Iterationen noch nicht erreicht haben.

Der Grenzwert in Schritt 3 hängt von den Kanalbedingungen ab (also von der Änderungswahrscheinlichkeit des Kanals hängt). Gallager zeigte diesen optimalen Grenzwert für reguläre LDPC-Codes. Es gibt einen anderen bequemen Weg, einen Grenzwert zu erhalten, der sich an die Kanaleigenschaften effektiv anpasst: bei diesem Weg muss der Schritt 3 verändert werden, indem der Grenzwert mit dem maximalen Wert in der Sequenz gleichgesetzt wird, dann werden alle Elemente in f identifiziert, die dem Grenzwert entsprechen und alle mit diesen Elementen korrespondierenden Bits der Sequenz $\boldsymbol{r}$ umgekehrt.

4.9 Der Summen-Produkt-Algorithmus (SPA)

Der Summen-Produkt-Algorithmus ist ein allgemeiner Algorithmus, der beinahe-optimale Leistung über eine breite umfangreiche Klasse von Kanälen bietet. Der SPA vermeidet es, Normalisierungen berechnen zu müssen, welche wir beim weichen Decodierungsalgorithmus für den BSC brauchen, um sicherzustellen, dass $r_{mn}(0) + r_{mn}(1) = 1$ and $q_{mn}(0) + q_{mn}(1) = 1$. Der Nachteil des SPA ist, dass er

nur auf binäre Codes anwendbar ist, während der weiche Decodierungsalgorithmus bei binären und nicht-binären Codes angewendet werden kann.

Bei guten Codes ist der SPA in der Lage, ein unkorrigiertes Code-Wort mit einer Wahrscheinlichkeit nahe an 1 zu entdecken. Der SPA geht davon aus, dass die ausgetauschten Nachrichten statistisch unabhängig sind über den gesamten Decodierungsprozess. Die Annahme der Unabhängigkeit träte in dem Fall zu, wenn der Tanner-Graph keine Zirkel aufweisen würde. Außerdem würde der SPA in diesem Fall exakte LLRs (engl:log-likelihood ratio=LLR) erzielen. Bei einem Graphen mit der Taillenweite γ ist die Annahme der Unabhängigkeit zutreffend bis zur $\frac{\gamma}{2}$-ten Iteration, nach dieser $\frac{\gamma}{2}$-ten Iteration fangen die Nachrichen an, in den Zirkel des Graphen um sich selbst zu kreisen.

Der SPA leidet an Fehlerraten-Floors, die auf trapping sets in LDPC-Codes oder eine kleine Minimaldistanz zurückzuführen sind, aber es gibt Berechnungsmethoden für den SPA, die zu niedrigeren Fehlerraten-Floors führen, wobei die Leistung dieser Methoden schlechter ausfällt (gemessen in dB). Eine der Methoden wird als min-sum-Decodierung bezeichnet, sie ist SPA ähnlich und im BEC-Kanal identisch.

Als Spezialfall betrachten wir zuerst den binären AWGN-Kanal. Nehmen wir an, dass das empfangene Code-Wort $\boldsymbol{r}$ ist und das übertragene Code-Wort $\boldsymbol{c}$. Wir sind interessiert an der a posteriori Wahrscheinlichkeit (engl:aposteriori probability=APP), dass ein einzelnes Bit im übertragenen Code-Wort $\boldsymbol{c}$ bei gegebenem empfangenem Wort $\boldsymbol{r}$ gleich 0 ist. Das APP-Verhältnis (auch heißt the likelihood-/Wahrscheinlichkeitsverhältnis LR genannt) ist gegeben durch

$$l(c_n|\boldsymbol{r}) = \frac{P(c_n = 1|\boldsymbol{r})}{P(c_n = 0|\boldsymbol{r})} \tag{4.65}$$

Das numerisch stabilere log-APP-Verhältnis (auch genannt log-likelihood-Verhältnis) ist gegeben durch.

$$\lambda(c_n|\boldsymbol{r}) = log\frac{P(c_n = 1|\boldsymbol{r})}{P(c_n = 0|\boldsymbol{r})} = log\frac{P(c_n = 1|r_n, \{r_i, i \neq n\})}{P(c_n = 0|r_n, \{r_i, i \neq n\})} \tag{4.66}$$

Unter Nutzung der Bayesschen Regel $P(A|B) = P(A,B)/P(B)$ können wir den Numerator umschreiben als

$$P(c_n = 1|r_n, \{r_i, i \neq n\}) = \frac{P(r_n, c_n = 1, \{r_i, i \neq n\})}{P(r_n, \{r_i, i \neq n\})}$$

Wenn wir dann die Regel $P(A,B,C) = [P(A \mid B,C)P(B,C)]/P(B)$ anwenden, haben wir

$$P(c_n = 1|r_n, \{r_i, i \neq n\}) = \frac{P(r_n|c_n = 1, \{r_i, i \neq n\})P(c_n = 1, \{r_i, i \neq n\})}{P(r_n|\{r_i, i \neq n\})P(\{r_i, i \neq n\})}$$

Wir wissen auch, dass wenn die Quelle gedächtnislos ist, haben wir $P(r_n|c_n = 1, \{r_i, i \neq n\})$=$P(r_n|c_n = 1)$ (das bedeutet, dass Bit r_n von den Bits $\{r_i, i \neq n\}$ bei gegebenem Bit c_n unabhängig ist) was zur Gleichung führt

$$P(c_n = 1|r_n, \{r_i, i \neq n\}) = \frac{P(r_n|c_n = 1)P(c_n = 1, \{r_i, i \neq n\})}{P(r_n|\{r_i, i \neq n\})P(\{r_i, i \neq n\})} \quad 4.67$$

And by using the Bayes' rule wir haben

$$\frac{P(c_n = 1, \{r_i, i \neq n\})}{P(\{r_i, i \neq n\})} = P(c_n = 1|\{r_i, i \neq n\}) \quad 4.68$$

Durch Einsetzen der Gleichung 4.68 in 4.67 bekommen wir

$$P(c_n = 1|r_n, \{r_i, i \neq n\}) = \frac{P(r_n|c_n = 1)P(c_n = 1|\{r_i, i \neq n\})}{P(r_n|\{r_i, i \neq n\})} \quad 4.69$$

Verfahren wir in ähnlicher Weise mit dem Denominator in Gleichung 4.66, haben wir

$$P(c_n = 0|r_n, \{r_i, i \neq n\}) = \frac{P(r_n|c_n = 0)P(c_n = 0|\{r_i, i \neq n\})}{P(r_n|\{r_i, i \neq n\})} \quad 4.70$$

Durch Einsetzen der Gleichungen 4.69, 4.70 in Gleichung 4.66 erhalten wir

$$\lambda(c_n|\boldsymbol{r}) = log\frac{P(r_n|c_n = 1)P(c_n = 1|\{r_i, i \neq n\})}{P(r_n|c_n = 0)P(c_n = 0|\{r_i, i \neq n\})} \quad 4.71$$

Wenden wir die Regel $\log(A.B) = logA + logB$ an, schreiben wir die letzte Gleichung als

$$\lambda(c_n|\boldsymbol{r}) = log\frac{P(r_n|c_n = 1)}{P(r_n|c_n = 0)} + log\frac{P(c_n = 1|\{r_i, i \neq n\})}{P(c_n = 0|\{r_i, i \neq n\})} \quad 4.72$$

Für einen Gaußschen Kanal $\sim\mathcal{N}(0, \sigma^2 = \frac{N_0}{2})$. wenn das übertragene Signal $\sqrt{E_b}$ ist (also wenn das Quellen-Bit 1 ist). Die Wahrscheinlichkeitsverteilungsfunktion des empfangenen Signals gegeben durch

$$P\left(r_n|c_n = \sqrt{E_b}\right) = \frac{1}{2\pi}e^{-\frac{1}{2\sigma^2}\left(r_n - \sqrt{E_b}\right)^2} \quad 4.73$$

Die Wahrscheinlichkeitsverteilungsfunktion des empfangenen Signals, wenn das übertragene Signal $-\sqrt{E_b}$ ist (also wenn das Quellen-Bit 0 ist), gegeben durch

$$P\left(r_n|c_n=-\sqrt{E_b}\right)=\frac{1}{2\pi}e^{-\frac{1}{2\sigma^2}\left(r_n+\sqrt{E_b}\right)^2} \tag{4.74}$$

In Abhängigkeit von den Gleichungen 4.73, 4.74 können wir schreiben

$$\begin{aligned} log\frac{P(r_n|c_n=1)}{P(r_n|c_n=0)} &= log\frac{P\left(r_n|c_n=\sqrt{E_b}\right)}{P\left(r_n|c_n=-\sqrt{E_b}\right)} \\ &= log\frac{\exp\left(-\frac{1}{2\sigma^2}\left(r_n-\sqrt{E_b}\right)^2\right)}{\exp\left(-\frac{1}{2\sigma^2}\left(r_n+\sqrt{E_b}\right)^2\right)}=\frac{2\sqrt{E_b}}{\sigma^2}r_n \end{aligned} \tag{4.75}$$

Aber da wir LDPC-Codierung für die übertragenen Bits im Gausschen Kanal nutzen, wird die Energie pro Bit nicht $\sqrt{E_b}$ ist, sondern $\sqrt{E_C}$ (was kleiner ist als $\sqrt{E_b}$). Unter Nutzung dieser Tatsache können wir schreiben

$$log\frac{P(r_n|c_n=1)}{P(r_n|c_n=0)}=\frac{2\sqrt{E_C}}{\sigma^2}r_n=L_cr_n \tag{4.76}$$

Wo L_c als Kanal-Zuverlässigkeit bezeichnet und der LLR-Wert L_cr_n aus dem Kanal-Sample r_n berechnet wird, so können wir Gleichung 4.72 in der Form schreiben

$$\lambda(c_n|\boldsymbol{r})=L_cr_n+log\frac{P(c_n=1|\{r_i,i\neq n\})}{P(c_n=0|\{r_i,i\neq n\})} \tag{4.77}$$

Wir bezeichnen L_cr_n als intrinsisch, da der LLR-Wert L_cr_n nur aus dem Kanal-Sample r_n berechnet wird. Den zweiten Term in 4.77 bezeichnen wir als extrinsisch, da er durch die anderen beobachteten Bits von $\{r_i,i\neq n\}$ bestimmt wird und auch durch die Code-Struktur (das heißt, durch die c-Knoten-Struktur des Codes).

Wir wissen, dass die Wahrscheinlichkeit $P(c_n=1|\{r_i,i\neq n\})$ entspricht, dass all die Paritätsprüfungen, die mit dem variablen knoten c_n verbunden sind (also alle Prüfknoten der Gruppe M_n), sind bei unberücksichtigung des Bits c_n gleich 1.

Die Wahrscheinlichkeit $P(c_n=0|\{r_i,i\neq n\})$ entspricht, dass all die Paritätsprüfungen, die mit dem variablen Knoten c_n verbunden sind (also alle Prüfknoten der Gruppe M_n), sind bei unberücksichtigung des Bits c_n gleich 0. Daher schreiben wir die letzte Gleichung in der Form

$$\lambda(c_n|\boldsymbol{r}) = L_c r_n + log\frac{P(z_{m,n} = 1\ für\ alle\ m \in M_n|\{r_i, i \neq n\})}{P(z_{m,n} = 0\ für\ alle\ m \in M_n|\{r_i, i \neq n\})} \quad 4.78$$

Wobei $z_{m,n}$ die Paritätsprüfung ist, die unter Nutzung des *m*ten c-Knoten mit c_n verbundenen Prüfknoten berechnet wurde. Das bedeutet

$$z_{m,n} = \sum_{i \in N_{m,n}} c_i \quad 4.79$$

Wo $N_{m,n}$ die Gruppe aller mit dem *m*ten c-Knoten verbundenen v-Knoten ist bis auf den v-Knoten c_n. Bei der Annahme, dass alle mit $z_{m,n}$ verbundenen Bits von den mit dem anderen Check $z_{m',n}$ verbundenen Bits unabhängig sind (das heißt, wenn unser Graph keine Zirkel aufweist), können wir die Gleichung 4.78 in der Form schreiben

$$\lambda(c_n|\boldsymbol{r}) = L_c r_n + log\frac{\prod_{m \in M_n} P(z_{m,n} = 1\ |\{r_i, i \neq n\})}{\prod_{m \in M_n} P(z_{m,n} = 0|\{r_i, i \neq n\})} \quad 4.80$$

Aus der Logarithmus-Regel ergibt sich

$$\lambda(c_n|\boldsymbol{r}) = L_c r_n + \sum_{m \in M_n} log\frac{P(z_{m,n} = 1\ |\{r_i, i \neq n\})}{P(z_{m,n} = 0|\{r_i, i \neq n\})} \quad 4.81$$

Nehmen wir den Vektor $d = [c_0, c_1, \dots, c_{|N_{m,n}|}]$ der v-Knoten im Check $z_{m,n}$, wobei $N_{m,n}$ wie vorher eine Gruppe von alle variablen Knoten ist, die mit dem m-ten c-knoten verbunden sind, bis auf den v-Knoten c_n, und $|N_{m,n}|$ ist die Anzahl der Elemente in der Gruppe $N_{m,n}$. Dann ist die Wahrscheinlichkeit, dass d eine gerade Anzahl Einsen enthält, gleich

$$\frac{1}{2} + \frac{1}{2}\prod_{l=0}^{|N_{m,n}|}(1 - 2P(c_l = 1)) \quad 4.82$$

Und die Wahrscheinlichkeit, dass d eine ungerade Anzahl Einsen enthält, ist

$$\frac{1}{2} - \frac{1}{2}\prod_{l=0}^{|N_{m,n}|}(1 - 2P(c_l = 0)) \quad 4.83$$

In Abhängigkeit von 4.79, 4.82 ist die Wahrscheinlichkeit, dass $z_{m,n} = 0\ |\{r_i, i \neq n\}$ gegeben durch

$$P\left(z_{m,n}=0\middle|\{r_i, i\neq n\}\right) = \frac{1}{2}+\frac{1}{2}\prod_{l=0}^{|N_{m,n}|}\left(1-2P(c_l=1|\{r_i, i\neq n\}\right) \quad 4.84$$

Durch Multiplikation der Gleichung mit 2 und durch das Ersetzen von $P\left(z_{m,n}=0\middle|\{r_i, i\neq n\}\right)$ durch $1-P\left(z_{m,n}=1\middle|\{r_i, i\neq n\}\right)$, erhalten wir

$$1-2P\left(z_{m,n}=1\middle|\{r_i, i\neq n\}\right) = \prod_{l=0}^{|N_{m,n}|}\left(1-2P(c_l=1|\{r_i, i\neq n\})\right) \quad 4.85$$

Wir können dies in eine LLR-Darstelung umwandeln, indem wir die leicht zu beweisende Beziehung für eine binäre Zufallsvariable mit den Wahrscheinlichkeiten p_1 und p_0 verwenden.

$$1-2p_1 = \tanh\left(\frac{1}{2}log\left(\frac{p_0}{p_1}\right)\right) = \tanh\left(\frac{1}{2}LLR\right) \quad 4.86$$

Folglich schreiben wir unter Nutzung der Gleichung 4.85 dann

$$tanh\left(\frac{1}{2}log\frac{P\left(z_{m,n}=0\middle|\{r_i, i\neq n\}\right)}{P\left(z_{m,n}=1\middle|\{r_i, i\neq n\}\right)}\right) = \prod_{l=0}^{|N_{m,n}|} tanh\left(\frac{1}{2}log\frac{P(c_l=0|\{r_i, i\neq n\})}{P(c_l=1|\{r_i, i\neq n\})}\right) \quad 4.87$$

Nehmen wir den Term

$$\lambda\left(z_{m,n}\middle|\{r_i, i\neq n\}\right) = log\frac{P(z_{m,n}=1\,|\{r_i, i\neq n\})}{P(z_{m,n}=0|\{r_i, i\neq n\})} \quad 4.88$$

In Abhängigkeit von den letzten zwei Gleichungen können wir die Gleichung 4.81 in der Form schreiben

$$\lambda(c_n|\boldsymbol{r}) = L_c r_n - 2\sum_{m\in M_n} tanh^{-1}\left(\prod_{j\in N_{m,n}} tanh\left(-\frac{\lambda(c_j|\{r_i, i\neq n\})}{2}\right)\right) \quad 4.89$$

Wir berechnen $\lambda(c_j|\{r_i, i\neq n\})$ auf die gleiche Art und Weise wie $\lambda(c_n|\boldsymbol{r})$, nehmen wir den Term

$$\eta_{m,n} = -2tanh^{-1}\left(\prod_{j\in N_{m,n}} tanh\left(-\frac{\lambda(c_j|\{r_i, i\neq n\})}{2}\right)\right) \quad 4.90$$

Folglich schreiben wir die Gleichung 4.89 als

$$\lambda(c_n|\boldsymbol{r}) = L_c r_n + \sum_{m\in M_n} \eta_{m,n} \quad 4.91$$

Man merke, dass, weil das Produkt in Gleichung 4.90 über die Menge $N_{m,n}$ ist, wurde die Nachricht von v-Knoten c_n an den c-Knoten *m* im Effekt heraussubtrahiert, um die extrinsische Information $\eta_{m,n}$ vom c-Knoten *m* an den v-Knoten c_n zu erhalten. Wir können auch sagen, dass $\lambda(c_n|\boldsymbol{r})$ die Nachricht ist, die der v-knoten c_n an seine c-knoten sendet.

Die letzten zwei Gleichungen bilden den Kern des SPA-Decoders. Das Standard-Stoppkriterium ist die Iteration zu stoppen, wenn $\hat{c}.H^T = 0$. Wobei $\hat{c}$ ein vorläufig decodiertes Code-Wort ist. Die Decodierung wird initialisiert, indem alle Nachrichten von v-Knoten gleichgesetzt werden mit

$$log\frac{P(r_n|c_n=1)}{P(r_n|c_n=0)} \quad 4.92$$

Das ist für alle *n*, für die $h_{ij} = 1$ in der Matrix *H* gilt.

Der Logarithmische-Wahrscheinlichkeitsverhältnisse-Decodierungs-Algorithmus für binäre LDPC-Codes:

Schritt1:
Dieser Schritt wird auch Initialisierungsschritt genannt. In diesem Schritt initialisiert *j* für jeden variablen Knoten $L_j = \lambda(c_j|\boldsymbol{r})$ nach Gleichung 4.92 für das geeignete Kanalmodell. Dann wird für alle i, j, für die $h_{ij} = 1$ gilt, die Nachricht von den v-Knoten an die c-Knoten $L_{j\to i} = L_j$ gesetzt.

Schritt2:
Dieser Schitt wird auch c-knoten Aktualisierungsschritt-Schritt genannt. In diesem Schritt berechnen wir die ausgehenden c-Knoten-Nachrichten $L_{i\to j}$ unter Nutzung derselben Gleichung 4.90 in der Form

$$L_{i\to j} = \eta_{m,n} = -2tanh^{-1}\left(\prod_{j'\in N_{m,n}} tanh\left(-\frac{L_{j'\to i}}{2}\right)\right) \qquad 4.93$$

Und dann übertragen wir jedes $L_{i\to j}$ zum variablen Knoten j.

Schritt3:
Dieser Schritt wird auch als Variable-Knoten-Aktualisierungsschritt bezeichnet. Hier werden die ausgehenden v-Knoten-Nachrichten $L_{j\to i}$ berechnet nach Gleichung

$$L_{j\to i} = L_j + \sum_{i'\in M_{n,m}} L_{i'\to j} \qquad 4.94$$

Und dann jedes $L_{j\to i}$ an den c-Knoten i übermittelt.

Schritt4:
Diesen Schritt nennen wir auch LLR total Schritt. In diesem Schritt berechnen wir für jeden v-Knoten j

$$L_j^{total} = L_j + \sum_{i\in N_m} L_{i\to j} \qquad 4.95$$

Schritt5:
Der Schritt wird auch als Stopp-Kriterium-Schritt bezeichnet. In diesem Schritt berechnen wir für jeden v-Knoten j das $\hat{c}$ wie folgt.

$$\hat{c} = \begin{cases} 0 & if\ L_j^{total} < 0 \\ 1 & else \end{cases} \qquad 4.96$$

Falls $\hat{c}.H^T = 0$ oder gleich der Maximalanzahl der Iterationen erreicht, wird der Vorgang beendet; andernfalls wird wieder zu Schritt 2 zurückgekehrt.

Das Update der Gleichung 4.93 enthält das Produkt und die Funktionen tanh und $\tanh^{-1}$, was dieses Update sehr kompliziert macht. Gallager schlug eine Verbesserung der Gleichungen wie folgt vor:

Wir schreiben die Nachricht $L_{j\to i}$ mit Vorzeichen und Betrag

$$L_{j\to i} = \alpha_{ji}\beta_{ji}$$
$$\beta_{ji} = \left|L_{j\to i}\right|, \alpha_{ji} = sign(L_{j\to i}) \quad 4.97$$

Zuerst multiplizieren wir die beiden Seiten der Gleichung 4.93 mit $tanh/2$ und setzen den Term $L_{j'\to i}$ mit Vorzeichen und Betrag ein (wie Gleichung 4.97). Dann erhalten wir

$$\tanh\left(\frac{1}{2}L_{i\to j}\right) = \prod_{j'\in N_{m,n}} \alpha_{j'i} \prod_{j'\in N_{m,n}} \tanh\left(\frac{1}{2}\beta_{j'i}\right) \quad 4.98$$

Nun multiplizieren wir beide Seiten dieser Gleichung mit $2tanh^{-1}$, so haben wir

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} . 2tanh^{-1}\left(\prod_{j'\in N_{m,n}} \tanh\left(\frac{1}{2}\beta_{j'i}\right)\right) \quad 4.99$$

Wenn wir die Regel $log^{-1}(log\ (x)) = x$ nutzen, haben wir

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} . 2tanh^{-1}log^{-1}\log\left(\prod_{j'\in N_{m,n}} \tanh\left(\frac{1}{2}\beta_{j'i}\right)\right) \quad 4.100$$

Nach Anwendung der Regel $\log(ab) = loga + logb$ erhalten wir

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} . 2tanh^{-1}log^{-1} \sum_{j'\in N_{m,n}} log\left(\tanh\left(\frac{1}{2}\beta_{j'i}\right)\right) \quad 4.101$$

Definieren wir die Funktion

$$\Phi(x) = -log\left(tanh\frac{x}{2}\right) = log\left(\frac{e^x+1}{e^x-1}\right) \quad 4.102$$

So können wir die Gleichung 4.101 in der Form schreiben

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} . -\Phi^{-1}\left(\sum_{j'\in N_{m,n}} -\Phi\left(\beta_{j'i}\right)\right) \quad 4.103$$

Aus der Gleichung von $\Phi(x)$ sehen wir jedoch, dass wenn x positiv ist, ist $\Phi^{-1}(x) = \Phi(x)$. Wir sehen aus der Gleichung 4.97, dass $\beta_{j'i}$ immer positiv ist, also können wir die letzte Gleichung schreiben als

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} \cdot \Phi\left(\sum_{j'\in N_{m,n}} \Phi(\beta_{j'i})\right) \qquad 4.104$$

Wir können diese Gleichung nutzen anstatt der Gleichung 4.93 in Schritt 2. Dann wird die Gleichung 4.95 in der Form

$$\begin{aligned} L_j^{total} = \alpha_j{}'\beta_j{}' &= L_j + \sum_{i\in N_m} L_{i\to j} \\ &= \alpha_j\beta_j + \sum_{i\in N_m} \prod_{j'\in N_{m,n}} \alpha_{j'i} \cdot \Phi\left(\sum_{j'\in N_{m,n}} \Phi(\beta_{j'i})\right) \end{aligned} \qquad 4.105$$

Wobei $\alpha_j{}'\beta_j{}'$ der Wert des variablen Knotens j in der neuen Iteration ist und $\alpha_j\beta_j$ der Wert in der vorhergehenden Iteration.

Gallager zeigte, dass wir die letzte Gleichung berechnen können durch Parallelberechnungen wie in Abb. 4.10. Diese Darstellung zeigt alle Operationen in der Gleichung, aber ohne Vorzeichen der Nachrichten. Wo es sich beim Decoder-Input um eine Logaritmische-Wahrscheinlichkeits-Verhältnisse-Nachricht (engl:log-likelihood message) handelt. Wir berechnen $\Phi(\beta_{j'i})$ in der ersten Reihe für nur ein Bit im Code-Wort. In den weiteren Reihen wir berechnen $\sum_{j'\in N_{m,n}} \Phi(\beta_{j'i})$. Die fortschreitenden Reihen stellen die restlichen Operationen in der Gleichung dar.

Die Autoren von [JDA+07], [JVSV03] machten vor, wie man eine Nachschlagetabelle (engl:look-up table) für die Funktion $\Phi(x)$ beim LDPC-Decoder nutzt. Es gibt auch andere Methoden, die Berechnungen für den log-SPA-Decoder zu vereinfachen, wie Box-Plus SPA-Decoder und Min-Sum Decoder, die weniger komplex als Box-Plus SPA, und identisch mit der log-SPA-Methode ist, wenn es sich beim Kanal um einen BEC handelt.

Die Änderung des SPA-Algorithmus durch Nutzung des BFC-Kanals: in diesem Falls ist $r_n \in \{0,1,e\}$ und es gibt drei Möglichkeiten für den Wert von $P(r_n|c_n = b)$ wo $b \in \{0,1\}$ gegeben sind durch

$$P(r_n|c_n = b) = \begin{cases} 1\ wenn\ r_n = b \\ 0\ wenn\ r_n = b^c \\ 0.5\ wenn\ r_n = e \end{cases} \qquad 4.106$$

Wobei b^c das Gegenstück von b darstellt, daraus und aus der Nutzung der Gleichung 4.92 ergibt sich, dass

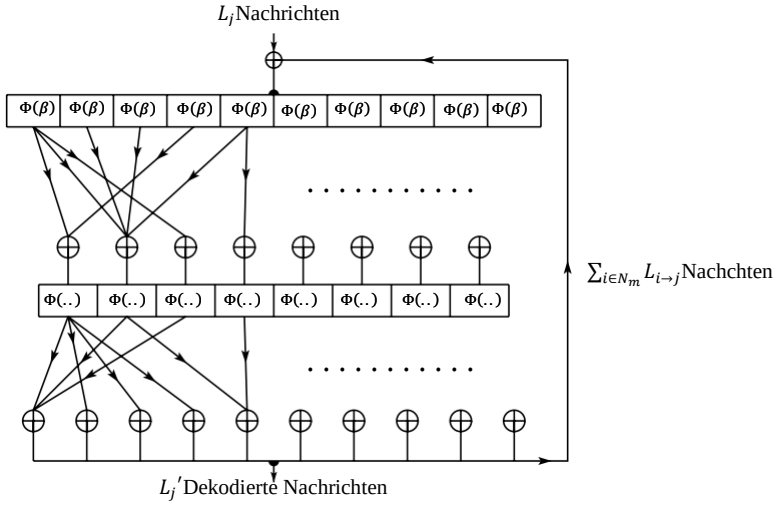


Abb. 4.10: Decodierungsstruktur des log-likelihood Decodierungsalgorithmus mit der Funktion Φ(x).

$$L(r_n|c_n) = \begin{cases} -\infty & wenn\ r_n = 0 \\ +\infty & wenn\ r_n = 1 \\ 0 & wenn\ r_n = e \end{cases} \qquad 4.107$$

Die Änderung des SPA-Algorithmus durch Nutzung des BSC-Kanals: in diesem Fall ist $r_n \in \{0,1\}$ und wir haben zwei Möglichkeiten für den Wert von $P(r_n|c_n = b)$ wo $b \in \{0,1\}$ gegeben sind durch

$$P(r_n|c_n = b) = \begin{cases} 1-\epsilon\ wenn\ r_n = b \\ \epsilon \quad wenn\ r_n = b^c \end{cases} \qquad 4.108$$

Wo ϵ die Änderungswahrscheinlichkeit des BSC-Kanals ist. Daraus und aus der Gleichung 4.92 folgt, dass

$$L(r_n|c_n) = \begin{cases} log\dfrac{\epsilon}{1-\epsilon}\ wenn\ r_n = 0 \\ log\dfrac{1-\epsilon}{\epsilon}\ wenn\ r_n = 1 \end{cases} \qquad 4.109$$

4.10 Der irreguläre LDPC-Code

Die Paritäts-Prüfmatrix H eines irregulären LDPC-Codes hat ebenfalls eine sehr dünn besetzte Matrix, aber die Spalten- und Zeilengewichte können von Spalte zu Spalte und von Zeile zu Zeile variieren.

Mit irregulären LDPC-Codes ist es möglich, sich Kapazitätsgrenzen viel mehr anzunähern als mit regulären LDPC-Codes.

Es ist üblich, den variablen Knoten und die Prüfknoten beim irregulären LDPC-Code durch Polynome, bezeichnet als λ(X) und ρ(X) festzulegen. Die Polynome λ(X) und ρ(X) stellen die Gradverteilung aus der "Kantenperspektive" (engl:Tanner graph's degree) dar.

$$\lambda(X) = \sum_{d=1}^{d_v} \lambda_d X^{d-1} \tag{4.110}$$

In der Polynomgleichung bezeichnet λ_d die Fraktion aller Kanten, die mit variablen Knoten d-ten Grades verbunden sind und d_v bezeichnet den maximal möglichen Grad von v-Knoten. Ähnlich, haben wir das Polynom

$$\rho(X) = \sum_{d=1}^{d_c} \rho_d X^{d-1} \tag{4.111}$$

ρ_d bezeichnet die Fraktion aller mit Prüfknoten d-ten Grades verbundenen Kanten. d_c bezeichnet den maximal möglichen c-Knoten Grad. Wir können die Gradverteilung auch aus einer "Knotenperspektive" darstellen mittels folgender Gleichungen.

$$\hat{\lambda}_d = \frac{\frac{\lambda_d}{d}}{\int_0^1 \lambda(X)dX} \tag{4.112}$$

$$\hat{\rho}_d = \frac{\frac{\rho_d}{d}}{\int_0^1 \rho(X)dX} \tag{4.113}$$

Das bedeutet $\hat{\lambda}_d$ ist die Fraktion aller v-Knoten vom Grad d. $\hat{\rho}_d$ ist die Fraktion aller c-Knoten vom Grad d. Die Anzahl von Kanten in einem Graphen ist gegeben durch

$$E = \frac{n}{\int_0^1 \lambda(X)dX} = \frac{m}{\int_0^1 \rho(X)dX} \tag{4.114}$$

wobei m die Anzahl der Zeilen der Paritäts-Prüfmatrix H, und n die Anzahl der Spalten der Matrix H ist. Die Anzahl variabler Knoten d-ten Grades ist gegeben durch

$$N_v(d) = E\frac{\lambda_d}{d} = \frac{n\frac{\lambda_d}{d}}{\int_0^1 \lambda(X)dX} \tag{4.115}$$

Die Anzahl von Prüfknoten *d*-ten Grades ist gegeben durch

$$N_c(d) = E\frac{\rho_d}{d} = \frac{m\frac{\rho_d}{d}}{\int_0^1 \rho(X)dX} \qquad 4.116$$

Wir wissen, dass die minimale Rate eines LDPC-Codes gegeben ist durch $1 - \frac{\mathrm{m}}{\mathrm{n}}$. Aus der Gleichung 4.114 schließen wir, dass die Coderatengrenze gegeben ist durch

$$R \geq 1 - \frac{m}{n} = 1 - \frac{\int_0^1 \rho(X)dX}{\int_0^1 \lambda(X)dX} \qquad 4.117$$

Der Autor von [LMSS01] zeigte, dass der Gewinn durch den irregulären LDPC-Code im Vergleich zu regulären LDPC-Codes bei 0.5 dB liegt. Tabelle 4.1 zeigt, dass für die gleiche Rate R=0.5 die unterschiedlichen Spalten-/Zeilengewichte unterschiedliche Leistungen zeigen, was bedeutet, dass wir zur Erstellung eines guten irregulären LDPC-Codes in der Lage sein müssen, das optimale Gewicht für den Code zu ermitteln. Tabelle 4.1 hängt davon ab, dass es sich beim Kanal um einen binären AWGN-Kanal mit Gaußschem Rauschen $\sim\mathcal{N}(0, \frac{N_0}{2})$ handelt. E_b ist die Energie pro übertragenes Bit, gegeben durch die Gleichung

$$E_b = \frac{Energieeinheiten\ pro\ Signal}{Anzahl\ bits/siknal} = \frac{E_S}{m} \qquad 4.118$$

Tabelle 4.1: Grenzwerte für verschiedene irreguläre LDPC mit Rate 0.5 bei binären AWGN Kanal.

Spalten Gewicht	Zeilen Gewicht	Rate	Grenzwert $\frac{E_b}{N_0}(dB)$	Kapazität (dB)	Lücke (dB)
3	6	0.5	1.1628	0.1871	0.9757
4	8	0.5	1.5944	0.1871	1.4073
5	10	0.5	2.0365	0.1871	1.8494

4.11 Density evolution

Density evolution ermöglicht die Analyse des Wahrscheinlichkeitsausbreitungs-algorithmus (engl:belief propagation algorithm), und hilft die Leistungsgrenzen von LDPC-Decodern zu verstehen. Es teilt uns einen Kanal-Grenzwert mit (wir bezeichnen diesen auch als Decodierungsgrenzwert oder Density-Evolution-Grenzwert), über dem

der Code gute Leistungen bringt (die Kommunikation ist zuverlässig) und unter dem die Fehlerwahrscheinlichkeit sehr hoch ist (die Kommunikation ist unzuverlässig). Dieser Kanalgrenzwert hat die Form E_b/N_0 für den binären AWGN und ϵ (Änderungswahrscheinlichkeit) für den BSC.

Dieser Kanal-Grenzwert hängt ab von den Eigenschaften des Kanals und den Ensemble-eigenschaften des LDPC-Codes (der Gradverteilungen des Codes $\lambda(X)$, $\rho(X)$, die durch die Gleichungen 4.110,4.111gegeben sind). Dieser einzelne Parameter erlaubt uns, die Leistung des LDPC-Codes mit der Grenze der Kanalkapazität zu vergleichen.

Die Berechnung der density evolution hängt ab von der Entwicklung der Wahrscheinlichkeitsdichtefunktion (engl: probability distribution function =PDF) der Nachrichten, die zwischen den v-Knoten und c-Knoten ausgetauscht werden, wobei die Nachrichten als Zufallsvariable und die Knoten als Quelle im Modell dargestellt werden.

Wir sollten anmerken, dass die Vorhersage der Leistung mittels density evolution schneller durchzuführen ist als eine Simulation der Leistung.

Die Autoren von [RU08], [RU01a] beschäftigten sich sehr ausführlich mit der Technik der density evolution.

Wir werden die wichtigsten Annahmen zur Berechnung der density evolution für den regulären LDPC-Code zusammenfassen.

---

1. All die Codes eines Ensemble weisen das gleiche Verhalten auf, so dass die Leistung eines spezifischen Codes (spezifische Paritäts-Prüfmatrix *H*) ist gegeben durch die durchschnittliche Ensemble-Leistung. In anderen Worten können wir sagen, wenn wir zwei LDPC-Codes haben mit den gleichen $\lambda(X)$, $\rho(X)$ aber mit unterschiedlichen Paritäts-Prüfmatricen *H*, dann weisen die beiden Codes gleiches Verhalten auf (im Sinne des Kanal-Grenzwerts).

2. Wir nehmen an, dass im Graphen kein Zirkel vorliegt, was oft zutrifft, wenn der Code sehr lang ist (*N* ist viel größer als *k*).

3. Wir nehmen auch an, dass der Kanal asymmetrisch ist, was für den binären AWGN bedeutet, dass

$$P\left(r_n \middle| c_n = \sqrt{E_b}\right) = P\left(-r_n \middle| c_n = -\sqrt{E_b}\right) \qquad 4.119$$

Wo r_n das empfangene Bit ist, c_n das übertragene Bit nach der LDPC-Codierung und $\sqrt{E_b}$ die Energie pro Bit (siehe Gleichung 4.118). Für den BSC ist die Annahme ähnlich und gegeben durch

$$P(r_n|c_n = 0) = P(r_n^{Co}|c_n = 1) \quad 4.120$$

Wobei r_n^{Co} das Gegenstück (engl:complement) von r_n darstellt.

Berechnen wir nun die density evolution:

Um den ganzen statistischen Decodierungsalgorithmus zu beschreiben, müssen wir die bedingt gemeinsame Wahrscheinlichkeitsdichtefunktion $P(\vec{m}_K|\vec{c})$ und auch deren Änderung mit der Iterationsanzahl k berechnen. Wobei $\vec{m}_K$ ein Vektor ist, der alle in der Iteration k gesendeten Nachrichten der v-Knoten enthält. Der Vektor $\vec{c}$ enthält die übertragenen Bits.

Dies führt dazu, dass um die allgemeine Funktion $P(\vec{m}_K|\vec{c})$ zu erhalten, wir alle möglichen übertragenen Sequenzen $\vec{c}$ berücksichtigen sollten. Wir wissen aber, dass die Anzahl aller möglichen übertragenen Sequenzen gegeben ist durch $Q = 2^N$, $N = |\vec{c}|$ was bedeutet, dass Q eine exponentielle Funktion übertragener Sequenzen ist. Daher haben wir bei einem großen N eine Situation ohne praktische Lösung. Das zweite Problem ist, dass die Anzahl der Pfade im Decodierungs-Graphen mindestens das zweifache der Code-Wort-Länge N beträgt, das bedeutet, dass die Anzahl aller möglichen Nachrichten im iterativen Decoder sehr groß ist, was zu einem großen Problem bei der Berechnung von $P(\vec{m}_K|\vec{c})$ führt, insbesondere weil die Funktion für jede Iteration k berechnet werden muss. Daher brauchen wir andere Ideen.

Die Idee der density evolution ist es, statt $P(\vec{m}_K|\vec{c})$ die Output-Verteilung einer einzelnen Nachricht $m_K^{v_i}$ vom v-Knoten i in der Iteration k zu berechnen. Wir wissen von der iterativen Decodierung des LDPC, dass die Nachricht des variablen Knotens von den Nachrichten abhängt, die von benachbarten Prüfknoten (Prüfknoten in $N_{m,n}$) in der Iteration $k-1$ empfangen wurden, und auch von der Nachricht des v-Knotens in der Iteration $k-1$, so können wir generell schreiben, dass

$$m_K^{v_i} = f_i(m_{K-1}^{v_i}, m_{k-1}^{c_{j_1}}, \dots, m_{k-1}^{c_{j_{|N_m|-1}}}) \quad 4.121$$

Aus 4.94 können wir die Funktion f_i schreiben als

$$m_K^{v_i} = m_{K-1}^{v_i} + \sum_{q \in N_{m,n}} m_{k-1}^{c_{j_q}} \quad 4.122$$

Für die erste Iteration $k = 0$ ist die Nachricht $m_{K-1}^{v_i}$ gegeben durch die Nachricht aus dem Kanal-Output, daher können wir für die erste Iteration schreiben:

$$m_1^{v_i} = m^{v_i} = m_0^{v_i} + \sum_{q \in N_{m,n}} m_{k-1}^{c_{jq}} \quad 4.123$$

Aus der letzten Gleichung können wir sagen, dass im Allgemeinen die PDF der Nachricht $m_K^{v_i}$ eine Funktion der gemeinsamen Verteilung aller Input-Nachrichten ist (der Nachrichten der Prüfcodes der Gruppe $N_{m,n}$), also können wir schreiben

$$\mathcal{P}_{m_k}^{v_i} = \Psi_i\left(\mathcal{P}_{m_{k-1}^{c_{j_1}}, \ldots, m_{k-1}^{c_{jq}}}\right) \quad 4.124$$

Wo $\mathcal{P}$ das Symbol der PDF-Funktion ist. Die Funktion Ψ ist abhängig von der Iteration Nummer i und hat generell keine geschlossene Form. Abhängig von der Annahme Nummer 2 weiter oben können wir davon ausgehen, dass die Nachrichten am Output der c-Knoten unabhängig sind, woraus sich ergibt, die PDFs dieser Nachrichten voneinander bedingt unabhängig sind, so können wir die Gleichung schreiben als

$$\mathcal{P}_{m_k}^{v_i} = \Psi_i\left(\mathcal{P}_{m_{k-1}}^{C_{j1}}, \ldots, \mathcal{P}_{m_{k-1}}^{C_{jq}}\right) \quad 4.125$$

Bei Gleichung 4.123 sehen wir, dass sie ein zeitlich invariantes System darstellt, was dazu führt, dass die PDF an ihrem Output (Output m^{v_i}) die Faltung der Input-Signale ($m_{k-1}^{c_{j_1}}, \ldots, m_{k-1}^{c_{j_{|N_m|-1}}}$) ist, also können wir schreiben

$$\mathcal{P}_{m_k}^{v_i} = \mathcal{P}_{m_{k-1}}^{v_i} * \mathcal{P}_{m_{k-1}}^{c_{j_1}} * \ldots * \mathcal{P}_{m_{k-1}}^{c_{jq}} \quad 4.126$$

Ausgehend davon, dass der Code regulär ist und von Annahme 3 oben, können wir annehmen, dass die PDFs der Nachrichten am Output der c-Knoten (die Nachrichten $m_{k-1}^{c_{j_1}}, \ldots, m_{k-1}^{c_{j_{|N_m|-1}}}$) identisch sind, also können wir die Gleichung 4.125 ohne Beachtung der Iterationsnummer und des Index des v-Knotens schreiben

$$\mathcal{P}^v = \Psi(\mathcal{P}^c) \quad 4.127$$

So ist die Gleichung 4.126 gegeben durch

$$\mathcal{P}^v = \mathcal{P}_0^v * [\mathcal{P}^c]^{|N_m|-1} \quad 4.128$$

Wir bezeichnen $\mathcal{P}^v$ als Initial-PDF des Kanal-LLR. Wir kennen die Regel, dass die Fourier-Transformation der Signalfaltung gleich ist dem Produkt der Fourier-Transformation der Signale, daher können wir die Gleichung 4.128 auf dem einfacheren Weg berechnen, dieser Weg ist

$$\mathcal{P}^v = \mathcal{F}^{-1}\left(\mathcal{F}(\mathcal{P}_0^v).[\mathcal{F}(\mathcal{P}^c)]^{|N_m|-1}\right) \quad 4.129$$

Diese Gleichung ist essentiell zu Berechnung der PDF des v-Knoten aus den PDFs der Nachrichten, die von den c-Knoten der Gruppe $N_{m,n}$ stammen.

Gehen wir von einem binären AWGN aus und davon, dass der Output-Vektor des LDPC ein Nullen-Code-Wort 0 sei, und dieser Vektor in der Binärphasenverschiebungscodierung (BPSK) gesendet wird. Bei der Annahme, dass die BPSK jede Null als +1 darstellt und jede 1 als -1, wird der Vektor am Output der BPSK ein Einsen-Code-Wort sein. Daher ist das übertragene Signal im Kanal ein Einsen-Code-Wort. Das führt dazu, dass nach einer unendlichen Anzahl von Iterationen k der Decodierungsfehler gegeben ist durch $\int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau$ (sieh Gleichung 4.92). Um nach einer unendlichen Iterationenanzahl keinen Decodierungsfehler zu haben, brauchen wir

$$\lim_{k\to\infty} \int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau = 0 \quad 4.130$$

Unter der Annahme, dass das übertragene Code-Wort nur aus Nullen besteht, haben wir keinen Decodierungsfehler, wenn

$$\lim_{k\to\infty} \int_{0}^{\infty} \mathcal{P}_k^v(\tau)d\tau = 0 \quad 4.131$$

Wir sollten anmerken, dass um die density evolution Function zu berechnen, die Annahme ausreicht, dass das Code-Wort nur aus Nullen oder nur aus Einsen besteht und wir andere möglichen Fälle von Code-Wörtern nicht näher zu betrachten brauchen, da wir angenommen haben, dass der Code linear und der Kanal symmetrisch ist.

Der Decodierungs-Grenzwert ist der Wert des Kanalparameters θ^*, so dass die iterative Decodierung eines bestimmten Ensemble eine Fehlerwahrscheinlichkeit aufweist, die gegen Null tendiert für alle Kanäle mit einer Genauigkeit höher als θ^* und weit von der Null entfernt ist für alle Kanäle mit der Genauigkeit kleiner als θ^*. Daher können wir die Gleichung von θ^* folgendermaßen schreiben

$$\theta^* = supremum\left(\theta: \lim_{k\to\infty} \int_{0}^{\infty} \mathcal{P}_k^v(\tau)d\tau = 0\right) \quad 4.132$$

Um $\mathcal{P}^v$ in Gleichung 4.129 zu erhalten, sollten wir $\mathcal{P}^c$ folgendermaßen berechnen:

Die Nachricht von c-Knoten an den v-Knoten ist in Gleichung 4.104 gegeben, aber wir schreiben sie betonungshalber wieder auf

$$L_{i\to j} = \prod_{j'\in N_{m,n}} \alpha_{j'i} \cdot \Phi\left(\sum_{j'\in N_{m,n}} \Phi\left(\beta_{j'i}\right)\right) \qquad 4.133$$

Der Wert $\alpha_{j'i}$ für unseren Fall eines binären AWGN ist entweder +1 oder -1, wir wollen nun die Multiplikation $\prod_{j'\in N_{m,n}} \alpha_{j'i}$ in der Gleichung in Summation in Modulo 2 umwandeln. Wir können dies durchführen, indem wir für $\alpha_{j'i}$ 1 einsetzen, wenn $\alpha_{j'i} = -1$. Wenn $\alpha_{j'i} = +1$ setzen wir für $\alpha_{j'i}$ 0 ein. Wenn die Funktion, in die eingesetzt wird, log zur Basis -1 ist, auch in Abhängigkeit von der Basiswechsel-Regel $\log_b x = \frac{\log_k x}{\log_k b}$, können wir leicht erkennen, dass $\log_{-1}\left(\alpha_{j'i} = 1\right) = \frac{\log_2(1)}{\log_2(-1)} = \frac{0}{\log_2(-1)} = 0$ und $\log_{-1}\left(\alpha_{j'i} = -1\right) = \frac{\log_2(-1)}{\log_2(-1)} = 1$ ist. So können wir in diesem Fall schreiben, dass

$$\prod_{j'\in N_{m,n}} \alpha_{j'i} = \sum_{j'\in N_{m,n}} \log_{-1}(\alpha_{j'i}) = S^{(C)} \qquad 4.134$$

Da $S^{(C)}$ das Vorzeichen der Nachrichten $L_{i\to j} = {L_{i\to j}}^{(C)}$ in 4.133 ist, können wir diese Nachrichten darstellen als einen Zweikomponenten-Zufallsvektor $\overline{L_{\iota\to J}}^{(C)}$. Die zwei Komponenten sind das Vorzeichen und der Betrag der Nachricht.

$$\overline{L_{\iota\to J}}^{(C)} = \left[S^{(C)}, \left|{L_{i\to j}}^{(C)}\right|\right] \qquad 4.135$$

Der Raumvektor des Zufallsvektors ist $\overline{L_{\iota\to J}}^{(C)} \in \{0{,}1\} \times \{0, \infty\}$. Wir können auch die Nachricht der v-Knoten darstellen als Zweikomponenten-Zufallsvektor in der Form

$$\bar{m}^{(v_i)} = \left[S^{(v_i)}, \left|m^{(v_i)}\right|\right] \qquad 4.136$$

Hier ist ebenfalls der Raumvektor des Zufallsvektors $\bar{m}^{(v_i)} \in \{0{,}1\} \times \{0, \infty\}$, also können wir die Gleichung 4.133 in Form Zufallsvektor schreiben als

$$\overline{L_{\iota\to J}}^{(C)} = \Phi^{-1}\left(\sum_{j'\in N_{m,n}} \Phi\left(\bar{m}^{(v_i)}\right)\right) \qquad 4.137$$

Wobei wir $\Phi(x)$ durch $\Phi^{-1}(x)$ ersetzen, da x hier immer positiv ist. Aus der letzten Gleichung könnten wir die PDF der Nachricht $\overline{L_{\iota\to J}}^{(C)}$ folgendermaßen berechnen, wenn uns die PDF der Nachricht $\bar{m}^{(v_i)}$ bekannt ist.

1. Wir berechnen die PDF des Zufallsvektors $\bar{z}_i = \Phi\left(\bar{m}^{(v_i)}\right) = -\ln\tanh\frac{\left(\bar{m}^{(v_i)}\right)}{2}$ in diesem Schritt. Aus der Gleichung 4.136 sehen wir, dass wenn $\bar{m}^{(v_i)}$ positiv ist, wir $\Phi\left(\bar{m}^{(v_i)}\right) = -\ln\tanh\frac{\left(m^{(v_i)}\right)}{2}$ schreiben können. Diese Funktion ist in der Form $Y(y) = g(X(x))$, die PDF dieser Funktion ist (siehe [Nol99]) gegeben durch die Regel $\mathcal{P}_Y(y) = \frac{\mathcal{P}_X(x)}{|g'(x)|_{x=g^{-1}(y)}}$, das führt in unserem Fall zu

$$\begin{aligned}\mathcal{P}(0,z_i) &= \left|\frac{dm^{(v_i)}}{dz_i}\right|.\mathcal{P}^v\left(m^{(v_i)}\right)\Big|_{m^{(v_i)}=\Phi^{-1}(z_i)} \\ &= \frac{1}{\sinh(z_i)}.\mathcal{P}^v(\Phi^{-1}(z_i)) \\ &= \frac{1}{\sinh(z_i)}.\mathcal{P}^v(\Phi(z_i))\end{aligned} \qquad 4.138$$

Wobei wir $\Phi(z_i)$ durch $\Phi^{-1}(z_i)$ ersetzt haben, weil z_i wie erwartet positiv ist. Wenn $\bar{m}^{(v_i)}$ negativ ist, können wir schreiben $\Phi\left(\bar{m}^{(v_i)}\right) = \ln\tanh\frac{\left(m^{(v_i)}\right)}{2}$. Auf die gleiche Weise bekommen wir

$$\mathcal{P}(1,z_i) = \frac{1}{\sinh(z_i)}.\mathcal{P}^v(-\Phi(z_i)) \qquad 4.139$$

2. In diesem Schritt werden wir die PDF des Zufallsvektors $\bar{w} = \sum_{i\in N_{m,n}}\bar{z}_i$ berechnen. Aus dem ersten Schritt wissen wir, dass $\bar{z}_i = \Phi\left(\bar{m}^{(v_i)}\right) \Longrightarrow \bar{w} = \sum_{i\in N_{m,n}}\Phi\left(\bar{m}^{(v_i)}\right)$. Diese Gleichung ist auch zeitinvariant, so ist die PDF von $\bar{w}$ gegeben durch

$$\mathcal{P}(\bar{w}) = \mathcal{P}(\bar{z}_1) * \mathcal{P}(\bar{z}_2) * \ldots\mathcal{P}\left(\bar{z}_{|N_{m,n}|}\right) \qquad 4.140$$

Wir erinnern uns auch, dass wir von einem regulären LDPC-Code ausgegangen sind und auch, dass kein Zirkel im Graphen vorliegt, was dazu führt, dass $\mathcal{P}(\bar{z}_1) = \mathcal{P}(\bar{z}_2) = \cdots = \mathcal{P}\left(\bar{z}_{|N_{m,n}|}\right) = \mathcal{P}(\bar{z})$. Also haben wir

$$\mathcal{P}(\bar{w}) = \mathcal{P}(\bar{z})^{*|N_{m,n}|} \qquad 4.141$$

Der leichtere Weg, diese Faltung zu berechnen, ist die zweidimensionale Fourier-Transformation

$$\mathcal{P}(0,\bar{w}) = \mathcal{F}^{-1}\left(\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(0,w)}\right)^{|N_{m,n}|} \qquad 4.142$$

$$\mathcal{P}(1,\overline{w}) = \mathcal{F}^{-1}\left(\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(1,w)}\right)^{|N_{m,n}|} \quad 4.143$$

$\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(0,w)}$ und $\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(1,w)}$ sind gegeben durch

$$\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(0,w)} = \mathcal{F}\{\mathcal{P}(0,z)\}_w + \mathcal{F}\{\mathcal{P}(1,z)\}_w \quad 4.144$$

$$\mathcal{F}\{\mathcal{P}(\bar{z})\}_{(1,w)} = \mathcal{F}\{\mathcal{P}(0,z)\}_w - \mathcal{F}\{\mathcal{P}(1,z)\}_w \quad 4.145$$

3. In diesem Schritt berechnen wir die PDF des Zufallsvektors $\overline{L_{\iota\to J}}^{(C)} = \Phi^{-1}(\overline{w}) = \Phi^{-1}\left(\sum_{j'\in N_{m,n}} \Phi\left(\overline{m}^{(v_i)}\right)\right)$. Indem wir die Gleichung wie in Schritt 1 ableiten, stellen wir fest, dass

$$\mathcal{P}\left(0, {L_{i\to j}}^{(C)}\right) = \frac{1}{\sinh\left({L_{i\to j}}^{(C)}\right)}.\mathcal{P}(0,\mathrm{z})\Big|_{z=\Phi\left({L_{i\to j}}^{(C)}\right)}, wenn\ S^{(C)} = 0 \quad 4.146$$

$$\mathcal{P}\left(1, {L_{i\to j}}^{(C)}\right) = \frac{1}{\sinh\left(-{L_{i\to j}}^{(C)}\right)}.\mathcal{P}(1,\mathrm{z})\Big|_{z=\Phi\left(-{L_{i\to j}}^{(C)}\right)}, wenn\ S^{(C)} = 1 \quad 4.147$$

Daher können wir durch Ausführung der drei genannten Schritte die PDF der Nachricht ${L_{i\to j}}^{(C)}$ berechnen bei Kenntnis der PDF $\mathcal{P}^v$ der v-Knoten-Nachrichten.

Die Zusammenfassung des density evolution Algorithmus

1. Man setze den Iterationen-Zähler $k = 0$ und den Kanalparameter θ (bzw. E_b/N_0 für den binären AWGN oder ϵ für den BSC) auf einen Wert, der unter dem Grenzwert θ^* liegen soll und berechne $\mathcal{P}_0^v$ aus den Kanaleigenschaften und dem empfangenen Vektor.

2. Setze zuerst die c-Knoten Nachrichten ${L_{i\to j}}^{(C)} = 0$, weil wir zuerst keine Nachrichten von den c-Knoten haben. So erhalten wir zuerst $\mathcal{F}(\mathcal{P}^c) = {L_{i\to j}}^{(C)} = 1$ und berechnen dann $\mathcal{P}^v$ aus 4.129

3. Wir gehen zur nächsten Iteration ($k = k + 1$), berechnen $\mathcal{P}^c$ (was nichts anderes ist als die PDF der Nachrichten ${L_{i\to j}}^{(C)}$) bei bekanntem $\mathcal{P}^v$ und führen die Schritte 1 – 3 in Seite 91 aus.

4. Berechne das neue $\mathcal{P}^v$ bei bekanntem $\mathcal{P}^c$ und $\mathcal{P}_0^v$.

5. Falls k kleiner ist als die Maximalanzahl von Iterationen und $\lim_{k\to\infty} \int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau$ sehr klein ist (normalerweise ist 10^{-6} ausreichend für die meisten Anwendungen), verringern wir den Kanalparameter θ um einen geringen Betrag (das bedeutet wir erhöhen die Fehlerwahrscheinlichkeit des Kanals) und gehen dann zu Schritt 2.

Falls $k < k_{max}$ und $\lim_{k\to\infty} \int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau$ nicht klein ist (größer als 10^{-6}) sollten wir die iterative Decodierung fortsetzen mit Schritt 3. Falls $\lim_{k\to\infty} \int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau$ nicht klein ist (größer als 10^{-6}) und die maximale Iterationsnummer erreicht ist ($k = k_{max}$), dann ist der Decodierungsgrenzwert für unseren Kanal und unsere Codes θ^*.

Beispiel für den binären AWGN:

Das $m_0^{v_i}$ (siehe Gleichung 4.123) ist für den binären AWGN durch Gleichung 4.76 gegeben, also haben wir $m_0^{v_i} = \frac{2\sqrt{E_C}}{\sigma^2} r_n$, wobei r_n der empfangene Vektor ist, gegeben durch $r_n = x_n + n$, wo x_n das Input-Signal ist, was bedeutet, dass x_n entweder $+\sqrt{E_C}$ oder $-\sqrt{E_C}$ sein kann, n ist das Gaußsche Rauschen $n \sim \mathcal{N}(0, \sigma^2 = \frac{N_0}{2})$. Nehmen wir an, dass ein Null-Bit abgebildet wird als eine Signalamplitude von $+\sqrt{E_C}$, und ein Eins-Bit als Signalamplitude von $-\sqrt{E_C}$. Wir nehmen in unserem Beispiel an, dass $x_n = +\sqrt{E_C}$, so schreiben wir

$$m_0^{v_i} = \frac{2\sqrt{E_C}}{\sigma^2} r_n = \frac{2\sqrt{E_C}}{\sigma^2}\left(\sqrt{E_C} + n\right) = \frac{2y}{\sigma^2}, wenn\ x_n = +\sqrt{E_C} \qquad 4.148$$

Es ist ersichtlich aus der Gleichung, dass die Nachrichten $m_0^{v_i}$ bedingt Gaußsche Nachrichten sind (auch wenn wir annehmen, dass $x_n = -\sqrt{E_C}$). Die Mittelwerte der Nachrichten $m_0^{v_i}$ sind gegeben durch

$$E\left(m_0^{v_i}\right) = \frac{2E(y)}{\sigma^2} = \frac{2E_C}{\sigma^2}\ , wenn\ x_n = +\sqrt{E_C}$$
$$E\left(m_0^{v_i}\right) = \frac{2E(y)}{\sigma^2} = \frac{-2E_C}{\sigma^2}\ , wenn\ x_n = -\sqrt{E_C} \qquad 4.149$$

Um die Varianz der Nachrichten $m_0^{v_i}$ zu berechnen, nutzen wir die Eigenschaft $var(aX + b) = var(aX) = a^2 var(X)$, wo a, b konstant sind, so haben wir

$$var\left(m_0^{v_i}\right) = var\left(\frac{2\sqrt{E_C}}{\sigma^2}\left(\sqrt{E_C} + n\right)\right) = \left(\frac{2\sqrt{E_C}}{\sigma^2}\right)^2 var(n) =$$
$$\left(\frac{2\sqrt{E_C}}{\sigma^2}\right)^2 \sigma^2 = \frac{4E_C}{\sigma^2}\ , wenn\ x_n = +\sqrt{E_C}\ or\ x_n = -\sqrt{E_C} \qquad 4.150$$

Wir sehen, dass die Varianz das Doppelte des Mittelwerts darstellt, so können wir sagen, dass $m_0^{v_i}$ Gaußsche Nachrichten sind, gegeben durch $m_0^{v_i} \sim \mathcal{N}(E(m_0^{v_i}), 2E(m_0^{v_i}))$, in anderen Worten ist $\mathcal{P}_0^v$ im density evolution Algorithmuns gleich $\sim\mathcal{N}(E(m_0^{v_i}), 2E(m_0^{v_i}))$. Ausgehend von diesem $\mathcal{P}_0^v$ und der Annahme, dass $x_n = +\sqrt{E_C}$ (das übertragene Signal im Kanal ist ein Einsen-Code-Wort, wenn $\sqrt{E_C} = 1$)

und mittels density evolution können wir die Änderung der PDF $\mathcal{P}_k^v$ der Nachrichten m^{v_i} sehen, wobei k die Iterationsnummer ist.

In Abb. 4.11 sehen wir, dass für das gleiche $\frac{\mathrm{E_b}}{\mathrm{N_0}}$ bei größerem k die Wahrscheinlichkeit, dass $\int_{-\infty}^{0} \mathcal{P}_k^v(\tau)d\tau = 0$ größer wird (was bedeutet, dass wir eine fehlerfreie Decodierung ausführen können), bei $\frac{E_b}{N_0} = 1.764$ dB liegt.

In Abb. 4.12 sehen wir die Änderung der Mittelwerte der Nachrichten m^v (es ist auch der Mittelwert der PDF $\mathcal{P}_k^v$) als Funktion der Iterationennummer k für verschiedene E_b/N_0. Wir sehen, dass für kleine E_b/N_0($E_b/N_0 \leq 1.76$ dB), der Mittelwert von m^v sich einer endlichen Grenze annähert, was bedeutet, dass wir keine fehlerfreie Decodierung durchführen können. Wenn E_b/N_0 steigt, geht der Mittelwert nach einigen Iterationen gegen Unendlich, während die Anzahl erforderlicher Iterationen mit steigendem E_b/N_0 absinkt.

Wir sehen auch, dass es einen Wert von E_b/N_0 gibt, über dem mit zuverlässiger Decodierung gerechnet werden kann (in unserem Beispiel liegt dieser Wert bei $E_b/N_0 \geq 1.764$ dB) und unterhalb dessen keine fehlerfreie Decodierung möglich ist. Dieser Wert wird als Grenzwert des Decoders bezeichnet (in unserem Beispiel ist der Grenzwert $E_b/N_0 = 1.764$ dB).

In Tab. 4.2 sehen wir, dass der Decodierungsgrenzwert $(E_b\ /\ N_0)^*$ dB ist (berechnet mittels des density evolution Algorithmus), wo $(E_b\ /\ N_0)^*$ der minimale Grenzwert für eine fehlerfreie Decodierung ist), und $(E_b\ /\ N_0)^{optimal}$ der optimale Grenzwert ist, um die Kanalkapazität eines binären AWGN zu erreichen. R ist die Codierungsrate des LDPC und d_v, d_c sind die Gradverteilungen der variablen bzw. der Prüfknoten. Wir geben diese Codierungsrate vor um die Kapazität des binären AWGN zu erfüllen.

Der Wert von $(E_b\ /\ N_0)^*$ wurde berechnet unter der Annahme, dass der Graph des LDPC-Code keinen Zirkel aufweist (wir arbeiteten mit dieser Annahme beim density evolution Algorithmus).

Wir sehen in Tab. 4.2, dass wir für das gleiche R (in anderen Worten für das gleiche $(E_b\ /\ N_0)^{optimal}$) verschiedene $(E_b\ /\ N_0)^*$ erzielen, je nach dem gewählten d_v, d_c. Zum Beispiel wenn $R = 0.5$ und $(E_b\ /\ N_0)^{optimal} = 0.187$ und $d_v = 3, d_c = 6$, liegt der Unterschied zwischen $(E_b\ /\ N_0)^{optimal}$ und $(E_b\ /\ N_0)^*$ bei 0.923dB, was kleiner ist als bei anderen Werten für d_v, d_c. Daher hängt, wie bereits gesagt, der Decodierungsgrenzwert nicht nur von $E_b\ /\ N_0$ ab, sondern auch von der Code-Struktur.

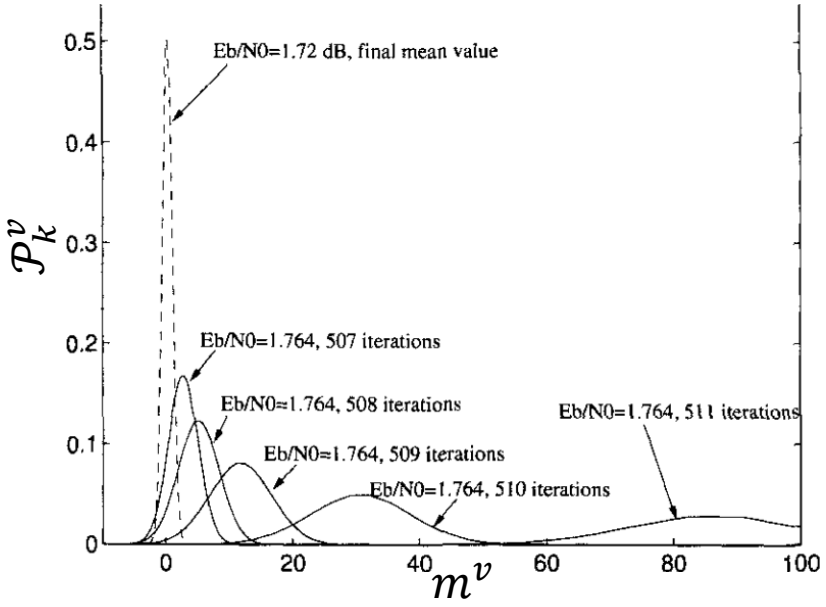


Abb. 4.11: Die PDF von m^v wenn die übertragene Nachricht ein Einsen-Code-Wort ist bei E_b/N_0 =1.72dB (final), und E_b/N_0 =1.764 dB (für verschiedene Iterationen). Quelle: [Moo05].

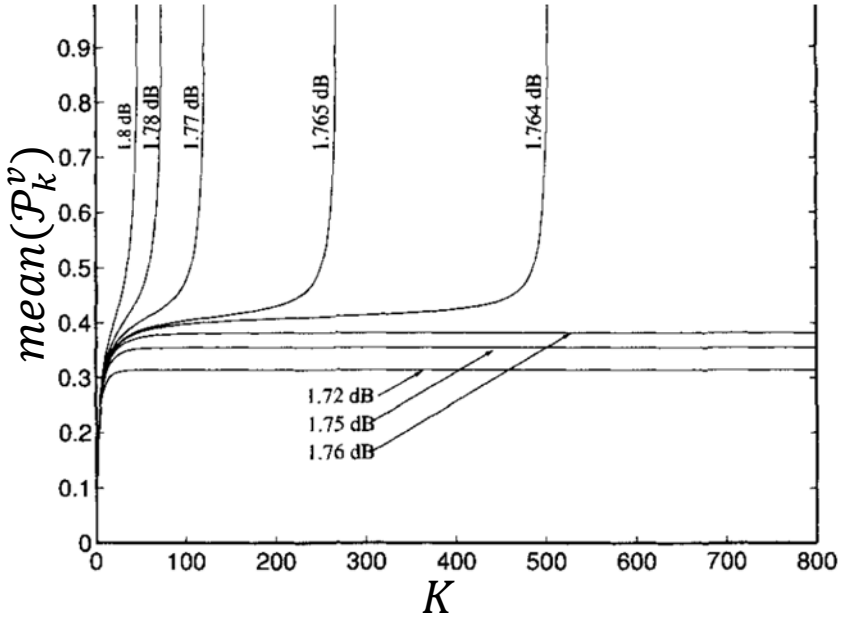


Abb. 4.12: Der Mittelwert der PDF $\mathcal{P}_k^v$ als Funktion der Iteration k bei verschiedenen Werten von E_b/N_0, wenn die übertragene Nachricht ein Einsen-Code-Wort ist. Quelle: [Moo05].

Wir sehen auch, dass es eine Tendenz zur Erhöhung von E_b / N_0 (also eine Verschlechterung) gibt, wenn die Rate des LDPC-Codes ansteigt. Wir sehen ebenfalls, dass in Allgemeinen wenn $d_v > 3$ ist, steigt der Grenzwert, was bedeutet, dass es immer besser ist, den LDPC-Code mit $d_v = 3$ zu konstruieren.

Tab. 4.2: Grenzwerte verschiedener LDPC-Codes für den binären AWGN-Kanal.

d_v	d_c	R	$(E_b\ /\ N_0)^*$	$(E_b\ /\ N_0)^{optimal}$
3	6	0.5	1.11	0.187
4	8	0.5	1.54	0.187
5	10	0.5	2.01	0.187
3	5	0.4	-0.078	-1.20
4	6	1/3	-0.094	-2.25
3	4	0.25	-2.05	-3.80

Beispiel für den BSC:

Wir betrachten einen binären symmetrischen Kanal mit Fehlerwahrscheinlichkeit ε, dessen Kapazität gegeben ist durch $1 - H(\varepsilon) = 1 + \varepsilon\log_2(\varepsilon) + (1-\varepsilon)\log_2(1-\varepsilon)$. Nehmen wir an, dass der Kanalinput ein regulärer LDPC-Code ist, dessen Rate ist gegeben durch $R = 1 - (d_v/d_c)$. Um den Code die Kanalkapazität ausfüllen zu lassen, sollten wir die Rate $R = 1 - H(\varepsilon)$ setzen. Mittels density evolution Algorithmus können wir das beste ε^* ausrechnen (im Sinne der maximalen Fehlerwahrscheinlichkeit, bei der noch eine fehlerfreie Codierung ausgeführt werden kann).

In Tab. 4.3 sehen wir, dass bei gleicher Codierungsrate (siehe unser Beispiel, wenn $R = 0.5$), verschiedene ε^* möglich sind, je nach gewähltem d_v, d_c. Wir sehen auch, dass es eine Tendenz zur Erhöhung von ε^*(also zur Verbesserung) gibt, wenn die Rate des LDPC-Codes niedriger wird. Angemerkt sei, dass ε^* die maximale Fehlerwahrscheinlichkeit ist, bei der wir noch eine fehlerfreie Decodierung ausführen können.

Tab. 4.3: Grenzwerte verschiedener LDPC-Codes für den BSC.

d_v	d_c	R	ε^*	ε
3	6	0.5	0.084	0.11
4	8	0.5	0.076	0.11
5	10	0.5	0.068	0.11
3	5	0.4	0.113	0.146
4	6	1/3	0.116	0.174
3	4	0.25	0.167	0.215

4.12 ASWG Paritätsmethoden mit LDPC bei Nutzung der Bitebenen-Faktorisierung

Wir werden in diesem Abschnitt sehen, wie Bitebenen-Faktorisierung dies ASWC-Paritätsmethode erleichtert und uns die Komplexität des Decoders erst bei Nutzung eines ML- und dann eines LDPC-Decoders anschauen. Wir werden auch sehen, wie bei dieser Methode die Wahrscheinlichkeiten berechnet werden, wenn das Modell des virtuellen Kanals ein Laplacesches Modell ist.

Wir werden zunächst einige Bezeichnungen erklären: ein quantisiertes Skalar, bezeichnet mit $\mathrm{Q} = \{\mathcal{P}, \mathcal{C}\}$ ist definiert durch eine Menge von Teilmengen $\mathcal{P} = \{\mathrm{P_j}\}_{\mathrm{j=0}}^{\mathrm{q-1}}$ und eine Code-Wort-Menge $\mathcal{C} = \{\mathrm{c_j}\}_{\mathrm{j=0}}^{\mathrm{q-1}}, \mathrm{c_j} \in \mathcal{R}$ ($\mathcal{R}$ ist die Menge der realen Zahlen). Die Teilmenge $\mathcal{P}$ ist definiert, dass $\mathrm{P_j}$ die Form $[\mathrm{a}, \mathrm{b})$, $\mathrm{P_j} \cap \mathrm{P_i} = \emptyset, \bigcup_{\mathrm{j=0}}^{\mathrm{q-1}} \mathrm{P_j} = \mathcal{R}$ hat. Das Code-Buch $\mathcal{C}$ ist so gewählt, dass $\mathrm{c_j} \in \mathrm{P_j}$. Die Quantisierung eines Skalars $\mathrm{x}, \mathrm{x} \in \mathcal{R}$ ist definiert als $\mathrm{Q(x)} = \mathrm{c_j}$, wenn $\mathrm{x} \in \mathrm{P_j}$. Um jedes $\mathrm{c_j}$ individuell darzustellen, brauchen wir einen $\mathrm{q'} = \log(\mathrm{q})$ binären Vektor $\mathrm{m(c_j)}$. Wir bezeichnen auch $\mathrm{m^i(c_j)}$ als Funktion des $\mathrm{i^{ten}}$ Bits von $\mathrm{m(c_j)}$.

Nehmen wir an, dass der Vektor $Q(x)$ aus n Elementen besteht, führt dies zu

$$Q(x) = \left[Q\big(V(1)\big), Q\big(V(2)\big), \dots, Q\big(V(n-1)\big)\right] \qquad 4.151$$

Wo jede Komponente $Q\big(V(i)\big)$ zum Code-Buch $\mathcal{C}$ gehört. Daher gehört das Code-Buch von $Q(x)$ zu $\mathcal{C}^n = \{\vec{c}_j\}_{j=0}^{q^n-1}$, wo $\vec{c}_j$ ein Vektor mit n Elementen ist, jedes Element gehört zum Code-Buch $\mathcal{C}$. Die mit $\mathcal{C}^n$ verbundene Teilmenge ist gegeben durch $\mathcal{P}^n = \{\vec{P}_j\}_{j=0}^{q^n-1}$, wo $\vec{P}_j$ ein Vektor mit n Elementen ist. Jedes Element gehört zu einem Element in der Gruppe $\mathcal{P}$.

Um einen linearen binären LDPC-Code zur Codierung von $Q(x)$ zu nutzen, müssen wir den Vektor $Q(x)$ binarisieren. Wenn $Q(x) = \vec{c}_j$, dann ist die binäre Darstellung von $Q(x)$ durch $m\big(Q(x)\big) = \left[m\left(\vec{c}_j(0)\right), m\left(\vec{c}_j(1)\right), \dots, m\left(\vec{c}_j(n-1)\right)\right]$ gegeben. Wir schreiben $m\big(Q(x)\big)$ in eine Matrix $\boldsymbol{B}_{n\times(q'-1)} = m\big(Q(x)\big)$, und konstruieren diese Matrix normalerweise in der Form $\boldsymbol{B} = \left[\boldsymbol{b}_0, \boldsymbol{b}_1, \dots, \boldsymbol{b}_{q'-1}\right]$, wo $\boldsymbol{b}_j$ der j^{te} binäre Spaltenvektor von $\boldsymbol{B}$ ist, die j^{ten} Bitebene in der binären Darstellung von $Q(x)$ repräsentierend.

Codierungsalgorithmus:

Um die Matrix $\boldsymbol{B}$ mittels LDPC zu codieren, ist, wie wir später in diesem Abschnitt sehen werden, zuerst $\boldsymbol{b}_0$ zu codieren und dann bis $\boldsymbol{b}_{q'-1}$, (wenn der Decoder zuerst das MSB des Vektors $Q(x)$ wiederherstellt, kann die nächste Bitebene (MSB-1) besser

wiedergegebeb werden und so weiter bis LSB). Der Hauptgrund, den Vektor x zu wählen aus n Elemente, ist dass alle Elemente im Vektor x beinahe die gleiche PDF haben, das führt dazu, dass bei der gemeinsamen Quantisierung dieser Elemente eher Verbesserungen eintreten als bei der getrennten Quantisierung der Samples.

Wir codieren jedes $\boldsymbol{b}_j$ mit der systematischen Generatormatrix (die Matrix hat die Form $\boldsymbol{G}_j = \left[\frac{\boldsymbol{I}_n}{\boldsymbol{P}_j}\right]$), wobei $\boldsymbol{I}_n$ eine $n \times n$ Einheitsmatrix (engl:identity matrix) ist und $\boldsymbol{P}_j$ eine $P_j \times n$ Paritätsmatrix. Also müssen wir zur Codierung der j^{ten} Bitebene $\boldsymbol{G}_j\boldsymbol{b}_j = \left[\frac{b_j}{\boldsymbol{P}_j\boldsymbol{b}_j}\right]$ berechnen. Die ersten n Bits des Vektors $\boldsymbol{G}_j\boldsymbol{b}_j$ sind der binäre Vektor $\boldsymbol{b}_j$. Zur Umsetzung der Idee aus Abschnitt 3.3.1.3 verwirft der Encoder die ersten n Bits (die systematischen Bits) und gibt die Paritätsbits $\boldsymbol{P}_j\boldsymbol{b}_j$ heraus. In anderen Worten braucht der Encoder nur $\boldsymbol{P}_j\boldsymbol{b}_j$ zu berechnen, das führt zu einer Komplexitätsreduktion am Encoder.

Das Output der gesamten Codierung (Codierung aller Bitebenen) ist die Menge der Paritätsvektoren $\{\boldsymbol{P}_j\boldsymbol{b}_j\}$,$0 \leq j < q'$. Die Codierungsrate des Vektors $Q(x)$ ist der Betrag aller Output-Paritäten ($|\{\boldsymbol{P}_j\boldsymbol{b}_j\}|$).

Wir werden nun diskutieren, wie $Q(x)$ mit der ASWC-Paritätsmethode decodiert werden kann (siehe Abschnitt 3.3.1.3) mit Hilfe eines LDPC-Decoders.

Maximum likelihood ohne LDPC-Decoder:

Nehmen wir an, dass die Zusatzinformation $\hat{x}$ (generell ist $\hat{x} \neq Q(x)$) am Decoder zur Verfügung steht. Der Decoder sollte die empfangen Paritätsbits nutzen, um Fehler in $\hat{x}$ zu korrigieren, das heißt, dass der Decoder $Q(x)$ aus $\hat{x}$ wiederherstellen muss.

Die Korrelation zwischen $Q(x)$ und $\hat{x}$ ist gegeben durch die Wahrscheinlichkeits-verteilungsfunktion $\mathcal{P}_{X|X'}(X|X')$. Wenn der Decoder diese Korrelation und die empfangenen Paritätsbits kennt, könnte er die Wahrscheinlichkeit von $Q(x) = \vec{c}_j$ berechnen mittels der Gleichung.

$$P\left(Q(x) = \vec{c}_j \middle| \hat{x}\right) = \int_{\vec{P}_j} \mathcal{P}_{X|X'}(X|X')\, dX \qquad 4.152$$

Wenn der Decoder keine Paritätsbits empfängt, ist die ML-Decodierung gegeben durch die Gleichung

$$\vec{V} = \arg\max_{\vec{c}_j \in \mathcal{C}^n} P\left(Q(x) = \vec{c}_j \middle| \hat{x}\right) \qquad 4.153$$

Bei Kenntnis der Paritätsvektoren $\{\boldsymbol{P}_j\boldsymbol{b}_j\}$ am Decoder verkleinert sich die Menge möglicher Rekonstruktionspunkte von $\mathcal{C}^n$ auf die Punkte $\vec{\Upsilon} \in \mathcal{C}^n$. Das $\vec{\Upsilon}$ erfüllt die Bedingungen für $\boldsymbol{P}_j m^j(\vec{\Upsilon}) = \boldsymbol{P}_j\boldsymbol{b}_j, 0 \le j < q'$, wobei $m^j(\vec{\Upsilon})$ die j^{te} Bitebene in der binären Darstellung von $\vec{\Upsilon}$ ist. Daher ist die ML-Decodierung in diesem Fall gegeben durch

$$\begin{aligned} &\vec{V} = \arg\max_{\vec{\Upsilon}\in\mathcal{X}} P\big(Q(x) = \vec{\Upsilon}\big|\hat{x}\big), where \\ &\mathcal{X} = \big\{\vec{\Upsilon}\colon \vec{\Upsilon} \in \mathcal{C}^n, \boldsymbol{P}_j m^j(\vec{\Upsilon}) = \boldsymbol{P}_j\boldsymbol{b}_j\big\}, 0 \le j < q' \end{aligned} \qquad 4.154$$

Die Differenz zwischen Gleichungen 4.154 und 4.153 ist der Suchraum. Durch geeignete Einschränkung des Suchraums ist es möglich $Q(x)$ aus $\hat{x}$ wiederherzustellen.

Wenn die Komponenten von X unabhängig und identisch verteilt sind (engl: independent identical distributed=i.i.d), können wir die PDF-Faktoren $\mathcal{P}_{X|X'}(X|X')$ schreiben als $\mathcal{P}_{X|X'}(X|X') = \prod_{i=0}^{n-1}\mathcal{P}_{x|x'}(X(i)|X'(i))$, und aus Gleichung 4.151 können wir schließen, dass $Q\big(X(i)\big)$ und $X'(i)$ sind i.i.d von $Q\big(X(j)\big)$ und $X'(j)$, $i \neq j$. Also können wir schreiben

$$P\big(Q(x) = \vec{c}_j\big|\hat{x}\big) = \prod\nolimits_{i=0}^{n-1}\mathcal{P}_{Q(X(i))|x'(i)}(\vec{c}_j(i)|X'(i)) \qquad 4.155$$

Es ist leicht zu erkennen, dass $\mathrm{P}\big(\mathrm{Q(x)} = \vec{\mathrm{c}}_\mathrm{j}\big|\hat{\mathrm{x}}\big) = \mathcal{P}_{\mathbf{B}|\mathrm{X}'}(\mathbf{B} = \widehat{\mathbf{B}}|\mathrm{X}')$, wobei $\widehat{\mathbf{B}} = \mathrm{m}(\vec{\mathrm{c}}_\mathrm{j})$. Wir bezeichnen auch $b_j(l)$ (das j^{te} Bit der binären Darstellung des l^{ten} Koeffizienten von $Q(x)$) als $b_{j,l}$. Die korrespondierende Quantität für X' ist $\hat{b}_{j,l} = \widehat{\boldsymbol{b}}_j(l)$. Daher ist $Q(x(l)) = \vec{c}_j(l)$ äquivalent zu $b_{j,l} = \hat{b}_{j,l}, 0 \le j < q'$. Also

$$\mathcal{P}_{Q(X(l))|x'(l)}\left(\vec{c}_j(l)\middle|X'(l)\right) = P(b_{j,l} = \hat{b}_{j,l}, 0 \le j < q'|X'(l)) \qquad 4.156$$

Bezeichnen wir $P\left(b_{j,l} = \hat{b}_{j,l}, 0 \le j < q'\middle|X'(l)\right)$ als $P\left(\{\hat{b}_{j,l}\}_{j=0}^{q'-1}|X'(l)\right)$. Weil jedes Bit $\hat{b}_{j,l}$ unabhängig ist von den Bits $\hat{b}_{i,l}$, wobei $0 \le i \le j-1$, können wir schreiben

$$P\left(\{\hat{b}_{j,l}\}_{j=0}^{q'-1}|X'(l)\right) = \prod\nolimits_{i=0}^{q'-1} P\left(\hat{b}_{j,l}|X'(l), \{\hat{b}_{j',l}\}_{j'=0}^{j-1}\right) \qquad 4.157$$

Durch Einsetzen von Gleichung 4.157 in Gleichung 4.156 und dann Gleichung 4.156 in 4.155 und Logarthmierung bekommen wir

$$\begin{aligned}&\arg max_{\vec{Y}\in\mathcal{X}}\, P\big(Q(x)=\vec{Y}\big|\hat{x}\big)\\&=\arg max_{\{\hat{b}_{j,l}:\hat{b}_{j,l}\in\mathcal{X}^b\}}\sum_{j=0}^{q'-1}\sum_{l=0}^{n-1}\log P\left(\hat{b}_{j,l}|X'(l),\{\hat{b}_{j',l}\}_{j'=0}^{j-1}\right)\end{aligned} \qquad 4.158$$

Wo $\mathcal{X}^b=\{\hat{b}_{j,l}:\boldsymbol{P}_j\hat{\boldsymbol{b}}_j=\boldsymbol{P}_j\boldsymbol{b}_j, 0\le j<q'-1\}$. Wie zuvor gesagt, ist die beste Decodierungsmethode um den Wert $\vec{Y}$ zu finden, welcher die letzte Gleichung erfüllt. Wir decodieren zuerst MSB, dann MSB-1 bis LSB. Bei dieser Vorgehensweise sind bei der Decodierung von j, die Bitebenen j', $j'<j$, bereits decodiert worden. Der Vorteil dieser Methode liegt darin, dass bei der Decodierung der j^{ten} Bitebene wir nicht alle Bedingungen in $\mathcal{X}^b$ erfüllen müssen, sondern nur die j^{te} Bedingung von $\mathcal{X}^b$. Diese neue Bedingung ist gegeben durch ${\mathcal{X}_j}^b=\{\hat{b}_{j',l}:\boldsymbol{P}_j\hat{\boldsymbol{b}}_j=\boldsymbol{P}_j\boldsymbol{b}_j\}$. Das heist wir können die richte Seite der letzten Gleichung schreiben als

$$\begin{aligned}&RHS\\&=\arg max_{\{\hat{b}_{0,l}:\hat{b}_{0,l}\in\mathcal{X}_0^b\}}\sum_{l=0}^{n-1}\log P\left(\hat{b}_{0,l}|X'(l)\right)+\cdots\\&+\arg max_{\left\{\hat{b}_{q'-1,l}:\hat{b}_{q'-1,l}\in\mathcal{X}_{q'-1}^b\right\}}\sum_{l=0}^{n-1}\log P\left(\hat{b}_{q'-1,l}|X'(l),\{\hat{b}_{j',l}\}_{j'=0}^{q'-1}\right)\end{aligned} \qquad 4.159$$

Die Optimierung eines jeden Terms in der letzten Gleichung ist nur über einer Bitebene und daher muss zur Optimierung nur über 2^n mögliche Werte gesucht werden, die der Bit-Vektor annehmen kann. Da wir $\log(q)$ Bitebenen haben, haben wir den Suchraum reduziert von q^n auf $2^n\log(q)$.

LDPC-Decoder bei Nutzung einer Bitebene Faktorisierung für Laplacesches virtuelles Rauschen:

Wir können den LDPC-Decoder nutzen, um die Maximierung der Gleichung 4.159 zu erzielen. Dies reduziert die Decodierungskomplexität (in anderen Worten die Suchzeit des Decoders) von einer Suche über 2^n Werte auf die lineare Zeit (die Komplexität der LDPC-Decodierung ist gegeben durch $O(n)$).

In meiner Arbeit gehe ich davon aus, dass das virtuelle Rauschen zwischen Quelle und Zusatzinformation eine Laplacesche Verteilung aufweist mit dem Mittelwert Null und der Standardabweichung $\frac{1}{\alpha}$, $P_N(n)=\frac{1}{2}\alpha\exp(-\alpha|X-Y)$.

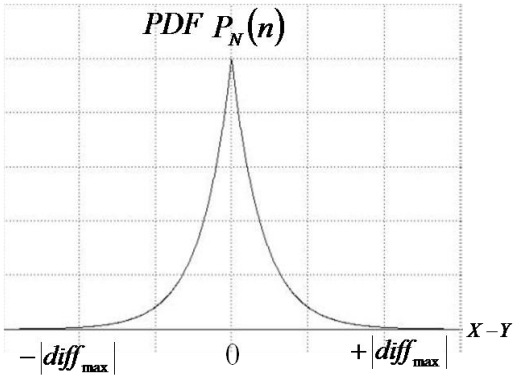


Abb. 4.13: Die Laplacesche PDF der Differenz zwischen Quelle X und Zusatzinformation Y.

Der LDPC-Decoder muss die Wahrscheinlichkeiten $P\left(\hat{b}_{j,l}|X'(l),\{\hat{b}_{j',l}\}_{j'=0}^{j-1}\right)$ berechnen (siehe letztes Abschnitt) um die Decodierung auszuführen und der Encoder muss die Parameter der Laplaceschen Verteilung von X kennen, um bei gegebener Zusatzinformation Y die Paritätsvektoren zu bestimmen und an den Decoder zu kommunizieren.

Bezeichnen wir die Quantisierung von X mit Q, und schreiben wir Q^i als das i^{te} Bit von Q. Die diskreten bedingten Wahrscheinlichkeiten von $P(Q|Y)$ werden errechnet, indem man die Wahrscheinlichkeiten von $P(X|Y) = P_N(n)$, für Werte von X, die in der gleichen Quantisierungsteilmenge liegen, addiert.

Wenn der LDPC-Decoder zu decodieren anfängt, sollte er zuerst die MSB von Q bei gegebener Zusatzinformation $Y = y$ rekonstruieren. Bei diesem Schritt kennt der Decoder die anderen Bits (die Bits MSB-1 bis LSB) noch nicht. Abb. 4.13 Zeigt die PDF der Differenz zwischen X und Y.

Die Zusatzinformation am Decoder sollte stets $|Y_{max}| = |X_{max}|$ erfüllen und der Mittelwert von X gleich dem Mittelwert von Y sein, was dazu führt, dass $|diff_{max}| = |Y_{max}| = |X_{max}|$. Die Wahrscheinlichkeit, dass MSB von Q gleich Null ist bei gegebener Zusatzinformation $Y = y$ ist gegeben durch

$$\begin{aligned} P(Q^{L-1} = 0|Y = y) &= \sum_{i=0}^{\left(\frac{2^{L-1}}{2}\right)-1} P(i|Y = y) \\ &= \sum_{i=0}^{\left(\frac{2^{L-1}}{2}\right)-1} P_N(i - y) \end{aligned} \qquad 4.160$$

$$P(Q^{L-1}=1|Y=y)=\sum_{i=\frac{2^{L-1}}{2}}^{2^{(L-1)}-1}P(i|Y=y)=\sum_{i=\frac{2^{L-1}}{2}}^{2^{(L-1)}-1}P_N(i-y) \qquad 4.161$$

Wobei die signifikanteste Bitebene von Q hier Q^{L-1} ist und die am wenigsten signifikante Bitebene Q^0. Wenn der LDPC-Decoder den Bit-Wert Q^b wiederherstellt und die Bit-Werte $b=b+1,\dots,L-1$ bereits decodiert sind, ist die Wahrscheinlichkeit von Q^b gegeben durch

$$\begin{aligned}P(Q^b=0|Y=y,Q^{b+1},\dots,Q^{L-1})&\\ =\sum_{i=0}^{2^b-1}P\left(\mathfrak{q}\left(x_p,2^{b+1}\right).2^{b+1}+i|Y=y\right)&\\ =\sum_{i=0}^{2^b-1}P_N\left(\mathfrak{q}\left(x_p,2^{b+1}\right).2^{b+1}+i-y\right)&\end{aligned} \qquad 4.162$$

$$\begin{aligned}P(Q^b=1|Y=y,Q^{b+1},\dots,Q^{L-1})&\\ =\sum_{i=2^b}^{2^{b+1}-1}P\left(\mathfrak{q}\left(x_p,2^{b+1}\right).2^{b+1}+i|Y=y\right)&\\ =\sum_{i=2^b}^{2^{b+1}-1}P_N\left(\mathfrak{q}\left(x_p,2^{b+1}\right).2^{b+1}+i-y\right)&\end{aligned} \qquad 4.163$$

Wobei x_p gegeben ist durch $x_p=\sum_{i=b+1}^{L-1}Q^i.2^i$, und die Funktion $\mathfrak{q}=(\mathrm{a},\mathrm{b})$ der Quotient aus den ganzen Zahlen a and b.

In Abb. 4.14 sehen wir $P_N(Q^b|Y=y,Q^{b+1},\dots,Q^{L-1})$, wo $Q=q$ das quantisierte Originalsymbol ist. Z_{max},Z_{min},Z_{mid} sind der maximale, der minimale und der mittlere Wert mit den gleichen vorher decodierten Bitebene-Werten wie $Q=q$. In Abb. 4.14 (a) liegt $Y=y$ für die Zusatzinformation zwischen Z_{min} und Z_{max}. In Abb. 4.14 (b) liegt y außerhalb des Bereiches zwischen Z_{min} und Z_{max}.

Die Fläche unter der PDF $P(X|Y=y)$ von Z_{min} bis Z_{mid} zeigt den Wert von $P(Q^b=0|Y=y,Q^{b+1},\dots,Q^{L-1})$ und die Fläche unter Z_{mid} bis Z_{max} zeigt den Wert von $P(Q^b=1|Y=y,Q^{b+1},\dots,Q^{L-1})$. Wir sollten anmerken, dass nur wenn $y=mean(Y)$, zeigt die PDF von $P_N(X|Y)$ das laplacesches Verhalten mit dem Mittelwert gleich y.

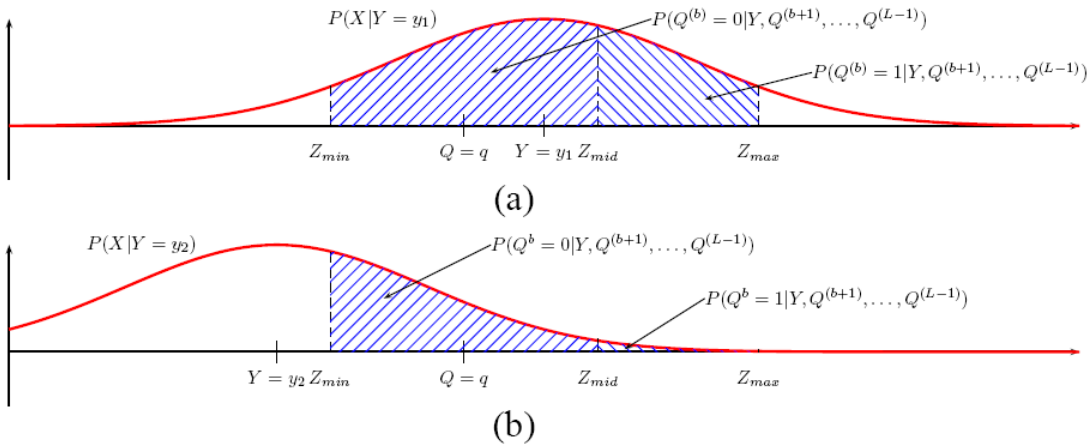


Abb. 4.14: Finden der bedingten Wahrscheinlichkeit von $Q^b = 0{,}1$ bei gegebenem bedingtem Wahrscheinlichkeitsdichte-Modell P(Q/Y) und gegebenen Werten aller zuvor decodierten Bitebenen $(Q^{b+1}, \ldots, Q^L)$.

Beispiel:

Nehmen wir an, dass die Zusatzinformation $y = 127$ ist und $L - 1 = 8$, die erste wiederhergestellte Bitebene von Q (MSB) ist $Q^{L-1} = Q^7 = 0$. Das führt zu $Z_{max} \equiv 01111111 \equiv 127$, $Z_{min} \equiv 00000000 \equiv 0$, $Z_{mid} \equiv 00111111 \equiv 63$. Der maximale Wert für Y könnte 255 sein und der minimale Wert 0. Es ist klar, dass die PDF $P(Q|Y)$ eine Laplacesche Funktion sein wird mit dem Mittelwert 127.

Mittels Gleichung 4.162 finden wir heraus, dass, um die nächste Bitebene ($b = 6$) zu korrigieren, wir $x_p = \sum_{i=b+1}^{L-1} Q^i . 2^i = \sum_{i=7}^{7} Q^i . 2^i = 0$, und $P(Q^6 = 0|Y = y = 127, Q^7 = 0) = \sum_{i=0}^{63} P(i|Y = y = 127) = \sum_{i=0}^{63} P_N(i - 127)$ erhalten. Das entspricht der Fläche unter der PDF in der Abb. 4.13 zwischen -127 und -64.

Auf dem gleichen Weg erhalten wir $P(Q^6 = 1|Y = y = 127, Q^7 = 0) = \sum_{i=64}^{127} P(i|Y = y = 127) = \sum_{i=64}^{127} P_N(i - 127)$. Das entspricht der Fläche unter der PDF in der Abb. 4.13 zwischen -63 und 0. Daher ist es klar, dass $P(Q^6 = 1|Y = y = 127, Q^7 = 0) > P(Q^6 = 0|Y = y = 127, Q^7 = 0)$.

5 State of the Art verteilte Bild- und Videocodierung

In diesem Abschnitt gibt es einen Überblick über state-of-the-art Artikel über verteilte Bildcodierung, und es werden Papers über verteilte Videocodierung vorgestellt, aus denen wir Ideen und Ansätze zur Forschung erhalten.

Generell ist das Interesse an verteilter Bildcodierung eher niedrig. Das allgemeine Interesse an verteilter Videocodierung ist allerdings höher. Der Grund dafür ist, dass bei der verteilten Videocodierung die Komplexitäteinsparung des Encoders sehr viel geringer als bei der verteilten Bildcodierung.

Die wichtigste Arbeit über verteilte Bildcodierung ist [ZLC08]. In dieser Arbeit wird angenommen, dass eine verzerrte Version des Originalbildes am Decoder vorhanden ist. Die Autoren designen einen Codec in der pixel-domain, um räumliche Redundanz auszunutzen und schlagen ein Markov Random Field-Modell (MRF) am Decoder vor, um räumliche Korrelation zu repräsentieren. Sie haben dabei dieses Modell am Decoder benutzt, um die Wahrscheinlichkeit der zu decodierenden Bits am LDPC-Decoder zu verändern, und gleichzeitig wurden die korrigierten Bits am Ausgang des LDPC-Decoders benutzt, um das MRF des Bildes zu verbessern. In diesem Paper wird LDPC auf der Bitebene der Pixel eingesetzt, und der LDPC-Decoder und das MRF tauschen iterativ Informationen aus, bis der Algorithmus konvergiert. Immer, wenn das MRF-Modell Informationen vom LDPC-Decoder erhält, wird das Bild geschätzt, und ein MRF wird aufgebaut, um die bedingte Wahrscheinlichkeit für jeden Pixel in Bezug auf seine Nachbarn zu bestimmen. Je höher die räumliche Korrelation des Bildes ist, umso stärker ist der Einfluss des MRF auf das LDPC-Decodieren. Es ist anzumerken, dass der Encoder in dieser Arbeit LDPC benutzt, um die Pixelwerte zu encodieren. Der Nachteil dieser Arbeit ist, dass die Autoren voraussetzen, dass eine verzerrte Version des Originalbildes am Decoder vorhanden ist. Diese Annahme ist in der Realität im Allgemeinen nicht gegeben.

Eine weitere Arbeit über verteilte Bildcodierung ist [CDB05]. In dieser Arbeit nutzen die Autoren verteilte Bildcodierung, um den Fehler an Kanten im Bild zu korrigieren. Sie verwenden nicht die wirklichen Pixelwerte zur Codierung der Seiteninformationen. Stattdessen benutzen sie die niederfrequenten Anteile des Bildes. Außerdem verwenden sie variierbarer Länge (engl:variable-length coset), um den Peak-SNR an Bildkanten zu erhöhen. Die Autoren nehmen in ihrer Arbeit an, dass die gemeinsame Verteilung, die den Eingangsdaten zugrunde liegt, am Encoder bekannt ist, um die Encoder-Bitrate zu bestimmen. Diese Annahme ist in der Realität nicht immer gegeben. Das vorgeschlagene Decodierschema enthält einen iterativen Prozess, um die Seiteninformation zu rekonstruieren, indem Wavelet-Zerlegung zusammen mit der korrekten Information der Nebeenklsse (engl:cosets) verwendet wird. Der Nachteil der Verwendung der Wavelets ist jedoch, dass die Wavelet-Transformation komplexer ist als die DCT. Daher ist der Encoder hier komplexer aufgebaut.

Die Autoren des Artikels [Liv02] zeigen ein verteiltes Quellencodierungsschema für korrelierte Bilder, das Modulo-Codierung von Pixelwerten sowie binäre und nichtbinäre Turbocodes nutzt, so dass höhere Bitrateneinsparungen als nur mit Modulo-Codierung praktisch verlustfrei erreicht werden können. Der Nachteil in dieser Arbeit ist, dass das ganze Bild am Eingang eines der beiden Encoder verfügbar ist. Dies entspricht nicht der Grundannahme über verteilte Codierung.

Im Artikel [WL09] schlagen die Autoren ein DCT-basiertes verteiltes Quellencodierungsschema niedriger Komplexität vor, um hyperspektrale Bilder zu komprimieren. Sie nutzen einen Ansatz, der auf der Zerlegung von Mengen von DCT-Koeffizienten basiert, um dann Bitebenen zu extrahieren; Danach wenden sie einen Slepian-Wolf Coder an, um die verteilte Bildcodierung durchzuführen. Ihre Arbeit beruht darauf, dass die Abhängigkeit von zwei zu encodierenden Teilinformationen in einem linearen Filter erster Ordnung dargestellt werden kann. Diese Annahme passt oft nur für die Hyperspectral Bilder. In ihren Ergebnissen zeigen sie, dass das vorgeschlagene Modell um bis zu 0.4 dB bessere Codierungswerte erreichen kann als verteilte Quellencodierung im diskreten Wavelet-Transformationsbereich.

Das Schema im Artikel [LVG09] verwendet einen *Expectation Maximization Algorithmus* in einem Slepian-Wolf-Decoder. Die Authentifizierung in ihrer Arbeit erfordert eine Slepian-Wolf-codierte Bildprojektion, die dem Decoder bekannt ist. Sie erweitern ihr Schema, um Bilder zu authentifizieren, die in den Bereichen Kontrast, Helligkeit und affinem Warping verändert wurden.

Der Artikel [YDD07] schlägt eine Methode zur Codierung von Multi-view Bildern vor, die sich auf "Multiterminal source coding" (MSC) stützt. Dank der separaten Codierung in MSC kann das vorgestellte Codierschema gute „Random access performance" erzielen, und die räumliche Redundanz kann selbst dann ausgenutzt werden, wenn die Encoder nicht miteinander kommunizieren können. Das Codierschema kann einfach auf N Ansichten erweitert werden, und die Kompressionsrate jeder Ansicht kann leicht angepasst werden, um den aktuellen Ressourcenanforderungen wie Bandbreite oder Speicher zu genügen. Die Ergebnisse des Experiments zeigen, dass die Kompressionsleistung besser ist als die des JPEG-En- und Decoders. Der Nachteil dieser Arbeit ist, dass die Autoren voraussetzen, dass die gemeinsame Verteilung, die den Eingangsdaten zugrunde liegt, am Decoder vorhanden ist. Diese Annahme trifft aber in der Realität im Allgemeinen nicht zu.

Es gibt eine Menge interessanter wissenschaftlicher Publikationen über verteilte Videocodierung, aber die wichtigste für uns ist [ASG03], in der die Autoren ein asymmetrisches Codierungsschema nutzen, in dem individuelle Frames eigenständig codiert (intraframe encoding), aber abhängig von anderen decodiert werden (interframe decoding). Eine Untermenge K von Frames wird als Keyframes markiert. Diese werden

mittels eines konventionellen intraframe codecs codiert. Zwischen den Keyframes liegen Wyner-Ziv-Frames, die intraframe-codiert werden, aber interframe decodiert.

Ein Wyner-Ziv-Frame, S, wird dabei wie folgt codiert: Wir quantisieren jeden Pixelwert des Frames mit einem uniformen scalaren Quantisierer mit 2^M Stufen, um den quantisierten Symbolstream q zu erzeugen. Diese Symbole nehmen wir, um ein langes Blocksymbol zu erzeugen, das dann zum Slepian-Wolf-Encoder geschickt wird. Der Slepian-Wolf-Encoder wird mit einem ratenkompatiblen punktierten Turbo-code (engl:rate compatible punctured turbo code) (RCPT) umgesetzt. Der RCPT wird mit einer Rückkopplung kombiniert und ermöglicht Ratenflexibilität, die essenziell für die Adaption an die immer wechselnden statistischen Verhältnisse zwischen Seiteninformation und dem zu codierenden Frame ist. Für jeden Wyner-Ziv-Frame nimmt der Decoder aufeinander folgende, vorher decodierte Key frames und evtl. vorher decodierte Wyner-Ziv-Frames, um die Seiteninformationen, S', aufzubauen, die eine Schätzung von S sind. Um diese Seiteninformation auszunutzen, nimmt der Decoder eine statistische Abhängigkeit zwischen S und S' an. Der vorgeschlagene Wyner-Ziv-Video-Coder verwendet rückgekoppelte Informationen vom Decoder, um die Anzahl der Paritätsbits zu bestimmen. Der Turbodecoder nutzt die Seiteninformation S' und die empfangene Untermenge an Paritätsbits, um den decodierten Symbolstrom S' zu erzeugen. Wenn der Decoder die Symbole nicht verlässlich decodieren kann, erzeugt er beim Encoder zusätzliche Paritätsbits über seine Rückkopplung.

In meiner Arbeit, die von dem o.g. Artikel über verteilte Videocodierung inspiriert wurde, verwende ich ein neues Modell der verteilten Bildcodierung. In meiner Arbeit teile ich die zu encodierende Bildinformation in zwei Teile. Der erste wird mittels Wyner-Ziv-codiert (mit LDPC), der zweite wird mittels H.264-Intra als “Key part” codiert und decodiert. Die beiden Teilinformationen werden in meiner Arbeit ebenfalls voneinander abhängig decodiert. Ich verwende LDPC statt Turbocodes weil es wie erwähnt bessere Ergebnisse erzielt, und ich benutze einen Rückkopplungskanal zwischen Encoder und Decoder, um adaptiv zu den wechselnden Statistiken zwischen Seiteninformation (Prädiktion des Wyner-Ziv-Anteils am Decoder) und dem zu codierenden Wyner-Ziv-Anteil zu bleiben. Ein weiterer Unterschied meiner Arbeit zur vorher erwähnten ist, dass ich nicht die Pixelwerte des Wyner-Ziv-Anteils encodiere, sondern stattdessen die ganzzahligen DCT-Koeffizienten des Wyner-Ziv-Anteils (es gibt einige andere Artikel, die sich mit verteilter Videocodierung beschäftigen und ebenfalls die DCT-Transformation des Wyner-Ziv-Frames codieren, wie der Artikel [BAP06]). In meiner Arbeit habe ich verschiedene Methoden verwendet, um das Bild in die zwei unterschiedlich zu codierenden Teile zu teilen, und ich habe ebenfalls verschiedene Methoden vorgestellt, um die Seiteninformation am Decoder zu erzeugen. Ich nehme in dieser Arbeit an, dass zwischen den beiden Encodern keine Kommunikation stattfindet. Außerdem nehme ich an, dass weder Encoder noch Decoder die Statistiken zwischen Seiteninformation (Wyner-Ziv-Anteil am Decoder) und dem ursprünglichen Anteil kennen. In meiner Arbeit wird der decodierte “Key part” und die Seiteninformation verwendet, um diese Statistiken vorherzusagen.

6 Implementierung und Bewertung der Methoden zur verteilten Bildcodierung

Das Ziel der vorliegenden Arbeit könnte in Kürze folgendermaßen definiert werden: Methoden zur verteilten Bildcodierung mit einer geringeren Codierungskomplexität als die der Intra-Codierung H.264 zu erreichen und zugleich das beste Rate-Verzerrung-Verhältnis zu erzielen.

In dieser Arbeit wird eine Wyner-Ziv-Bildercodierung, die so genannte verteilte Bidercodierung vorgestellt. Statt der Slepian-Wolf-Codierung wird diese Wyner-Ziv-Bildercodierung benutzt, weil in den meisten Fällen die zu codierenden Informationen (in unserem Fall: das Bild) bei verteilter Bild Codierung nicht verlustfrei zu codieren ist. Diese Informationen können nur dann verlustfrei codiert werden, wenn der virtuelle Kanal (siehe Abschnitt 3.1 und 3.2), der die zu codierende Information mit der vorhergesagten Information (das vorhergesagte Bild an dem Decoder) verbindet, dem Encoder bekannt ist und wenn die Eigenschaften des virtuellen Kanals (die Änderungswahrscheinlichkeit des Kanals) sich im Verlauf der Zeit nicht ändern. Damit der Encoder diese Eingenschaften kennt, muss dieser entweder mit den anderen Encoder kommunizieren – wobei es in verteilter Bild Codierung keine Kommunikation zwischen den Encodern gibt – oder muss der Decoder die Änderungswarscheinlichkeit für jede die zu codierende Information an den Encoder senden – was wiederum dazu führt, dass dann aber der Decoder mehr Informationbits sendet als der Entropy-Rate der Quelle entsprechend. Aus diesem Grund sind in vielen Anwendungen mit der verteilten Bildcodierung dem Encoder der virtuelle Kanal mit seinen Eingenschaften (seine Änderungswarscheinlichkeit usw.) nicht bekannt.

Es gibt verschiedene Methoden, um Wyner-Ziv-Codierung durchzuführen, aber wir nutzen die „ASWC mit der Paritätsmethode“ (siehe Abschnitt 3.3.1.2 und 4.12), weil es uns nicht wichtig ist, dass die Gesamtsumme der Raten der Encoder konstant bleibt, sondern dass das ganze Bild mit bester PSNR am Decoder wiederherstellt wird. Wir benutzen LDPC (engl: Low Density Parity Check Codes, siehe Kapitel 4), um die Paritätsbits in „ASWC mit der Paritätsmethode“ zu generieren. Deswegen können wir in unserer Arbeit Wyner-Ziv-Codierung als Wyner-Ziv-Codierung (LDPC) bezeichnen. Unser LDPC erzielt die wahrscheinlichsten fehlerkorrigierten Ergebnisse basierend auf weichen Input-Wahrscheinlichkeiten (siehe Kapitel 4). Der LDPC Decoder kann alle Fehler korrigieren, wenn manche Bedingungen erfüllt sind (siehe Kapitel 4). Deswegen können wir sagen, dass LDPC für die Information am Eingang der LDPC derselben Rolle entspricht wie eine Slepian-Wolf-Codierung.

Es werden drei Methoden in diesem Kapitel verwendet und anschließend bewertet. Wir nutzen in allen diesen drei Methoden die „ASWC mit der Paritätsmethode“, um die verteilte Wyner-Ziv-Bildercodierung auszuführen. Wir teilen das Bild bei allen unseren

Methoden in zwei Teile auf, in Bild I (Wyner-Ziv-Bild=WZ-Bild) und Bild II (Schlüssel-Bild=Sch_Bild); die Größe des WZ-Bildes sowie die des Sch_Bildes ensprechen jeweils der Hälfte des kompletten Bildes. Wir codieren das WZ-Bild mittels der WZ-Codierung (LDPC) und das Sch_Bild mittels der herkömmlichen Intra-Codierung H.264. Wir versuchen bei unseren drei Methoden, die Rechenlast der Codierung des Wyner-Ziv-Bildes zu reduzieren, aber das Rate-Verzerrung-Verhältnis des Intra-Codierung H.264 für das ganze Bild beizubehalten. Wir werden sehen, dass insbesondere bei der zweiten und dritten Methode diese Komplexität auf den Decoder verlagert wird und die Komplexität des Encoders bei unseren Methoden kleiner wird als die Komplexität der Intra-Codierung H.264. Jedoch können wir in keiner unserer Methoden das Rate-Verzerrung-Verhältnis der Intra-Codierung H.264 erreichen.

Der Wyner-Ziv Decoder nutzt Zusatzinformation, um das WZ-Bild wiederherzustellen. Diese Zusatzinformation ist nichts anderes als Intra-Vorhersage des WZ-Bild.

Wir nutzen für unsere Methoden die ganzzahlige DCT-Transformation (engl: integer DCT =IntDCT), da diese die Intra-Korrelation innerhalb eines Bildes nutzt und diese Transformation üblicherweise in traditioneller Bild- und Videocodierung (z. B. H.264) genutzt wird. Diese Transformation wird angewendet auf einen $n \times n$ Sample-Block des Bildes und sie dekorreliert die Intra-Korrelation zwischen den Samples innerhalb der Blöcke. So werden die Transformationskoeffizienten unabhängig von den benachbarten Samples des Bildes sein. Diese IntDCT stellt die Samples des Bildes in der Frequenz-Domäne dar und dies erlaubt, die Blockenergie in einer kleinen Anzahl der Koeffizienten zu konzentrieren. Der Koeffizient DC korrenspondiert mit der niedrigsten räumlichen Frequenz, er stellt den Mittelwert der Sample-Werte innerhalb des Blocks dar. Die Niedrigfrequenz-Koeffizienten schließen die meiste Sample-Block-Energie ein und die Amplitude der Hochfrequenz-Koeffizienten ist typischerweise nah der Null.

In unserer Arbeit wird eine Block-basierte Transformation verwendet in einem Szenario einer verteilten Bildercodierung mit dem gleichen Ziel, wie es für herkömmliche Codierungsschemata (Intra-Codierung in H.264, jpeg) benutzt wird. Wir benutzen in unserer Arbeit die ganzzahlige DCT-Transformation, da diese weniger komplex ist als DCT, was dabei hilft, die Codierungslast des Wyner-Ziv-Bildes zu reduzieren.

6.1 Annahmen für die Simulationen, Maße für Leistung und die Rate

Wie in der Literatur über verteilte Video-Codierung üblich, codieren wir in unserer Arbeit nur die Luminanzkomponente, und daher beziehen sich alle Maße nur auf die Luminanz. Trotzdem kann unsere Implementierung der verteilten Bildercodierung auf Chrominanzkomponente mit ähnlicher Leistung auch verwendet werden.

In unserer Arbeit hängt die Qualität des WZ-Bildes (ein Teil des ganzen Bildes) stark von der Qualität des Schlüssel-Bildes (der Rest des Bildes) ab, da die Qualität der benachbarten Blöcke in einem Bild einen großen Einfluss auf die subjektiv wahrgenomme Qualität des von ihnen umgebenen Blocks hat. Aus diesem Grund werden wir nicht die Qualität des WZ-Bildes allein, sondern die des gesamten Bildes bewerten.

Aus unseren RD-Kurven (engl:Rate-Distortion=RD) können wir ersehen, wie die Rate für das Schlüssel-Bild (H.264 Intra Bits) und die Rate für das WZ-Bild (LDPC Parität-Bits + andere übertragene Bits) sich auf die Qualität auswirken. Alle Maße beziehen sich sowohl auf die Schlüssel-Bild- als auch auf die WZ-Bild-Bits, da diese miteinander auf die Gesamtqualität des Decodierten Bild Einfluss nehmen.

In unserer Arbeit betrachten wir fehlerfreie Kanäle, aber es ist bekannt, dass Verteilte-Codierung-Lösungen für diejenigen Anwendungen von speziellem Interesse sind, bei denen der Kanal fehlerbehaftet ist.

Wir nutzen PSNR, um die objektive Qualität für die Luminanzwerte zu messen, dies ist gegeben durch

$$PSNR_{dB} = 10 \log_{10} \frac{(2^n - 1)^2}{MSE} \qquad 6.1$$

Dabei ist n die Anzahl der Bits pro Bilder-Sample, was bedeutet, dass für ein Luminanzbild n gleich 8 ist, weil der Luminanzwert zwischen 0 und 255 liegen kann. $(2^n - 1)^2$ ist das Quadrat des höchst möglichen Luminanzwertes im Bild, MSE ist der mittlere quadratische Fehler zwischen dem ganzen Originalbild und dem decodierten Bild am Decoder. Für ein und dasselbe Bild zeigt ein hoher PSNR eine höhere, ein niedriger PSNR eine schlechtere Qualität, aber ein bestimmter PSNR kann nicht zwangsläufig mit einer subjektiven Qualitätshöhe gleichgesetzt werden. Das Maß für PSNR ist dB.

Wir nutzen die H.264/AVC Video-Codierung im Hauptprofil (engl: Main profile) ohne Einsatz der zeitlichen Redundanz (H.264 Intra-Coding), um Schlüssel-Bild zu codieren. Die verwendete H.264 Intra-Coding Software codiert sowohl die Luminanz- als auch die Krominanz-Werte der Bilder. In unserer Arbeit interessieren uns jedoch nur die Krominanzwerte. Aus diesem Grund ändern wir alle Krominanzwerte des Schlüssel-Bildes auf den Wert 128 und reduzieren damit auch die Bitrate bei der Benutzung der H.264 Intra-Coding. Der Grund für die verringerte Bitrate ist, dass der Prädiktionsfehler bei der Codierung der Krominanzwerte sehr klein wird und alle Blöcke im Schlüssel-Bild den gleichen Intra-Vorhersage-Modus (engl:intra prediction mode) haben.

Die Kosten für die Codierung des gesamten Bildes werden in Bit pro Pixel gemessen (für das ganze Bild und nicht nur für das WZ-Bild).

Wir gehen in unserer Arbeit davon aus, dass die Wahrscheinlichkeitsdichtefunktion des virtuellen Kanals zwischen dem WZ-Bild-Koeffizienten und dessen vorhergesagten Bild-Koeffizienten einer Laplacesche Verteilung mit dem Mittelwert Null entspricht.

Wir werden die besten verfügbaren Quantisierungsebenen in der ganzzahligen DCT nutzen (Quantisierungsparameter *qp*=0), um für das WZ-Bild die bestmögliche Qualität ohne Block-Artefakte zu erzielen.

In unserer Arbeit verwenden wir 4 Testbilder (siehe Abb. 6.1). Die Bilder sind jeweils 256×256 Pixel groß. Das glatteste Bild ist *Haus* (die Niedrigfrequenz-Koeffizienten haben eine sehr hohe Energie im Vergleich zu den Hochfrequenz-Koeffizienten); die nächstglattesten Bilder sind dann in der Reihenfolge: *Lena*, *Barbara*, *Baboon*.



Abb. 6.1: Die Originalbilder *Haus*, *Lena*, *Barbara*, *Baboon* von links nach rechts und von oben nach unten.

Die logarithmischen Leistungsdichtespektren (LDS) der Bilder sind gegeben in Abb. 6.2. LDC ist die fouriertransformierte Autokorrelationfunktion des Bildes und beschreibt die Verteilung der Leistung im Frequenzbereich. Die Amplitude in Abb. 6.2 wird berechnet durch die Gleichung $A = \log_e \frac{1}{M \times N} |FFT_2(z)|^2$, in der M,N die Breite und die Länge des Bildes z sind; FFT_2 ist eine zweidimensionale Schnelle-Fourier-Transformation (engl:Fast Fourier transform =FFT). Die μ-Achse und ν-Achse stellen jeweils die Frequenz in waagerechter sowie senkrechter Richtung des Bildes dar. Das LDS zeigt eine Spitze bei $\mu = \nu = 0$. Wir sehen, dass das LDS des Bildes *Haus* in diagonaler Richtung einen hohen Betrag aufweist, obwohl das Bild *Haus* das glatteste Bild im Vergleich zu den anderen Bildern ist. Der Grund hierfür ist der, dass das Bild *Haus* in diagonaler Richtung starke Kanten hat. Wir sehen auch, dass das LDS des Bildes *Baboon* die geringste Leistung und somit im Allgemeinen keine große Korrelation zwischen den benachbarten Pixeln aufweist.

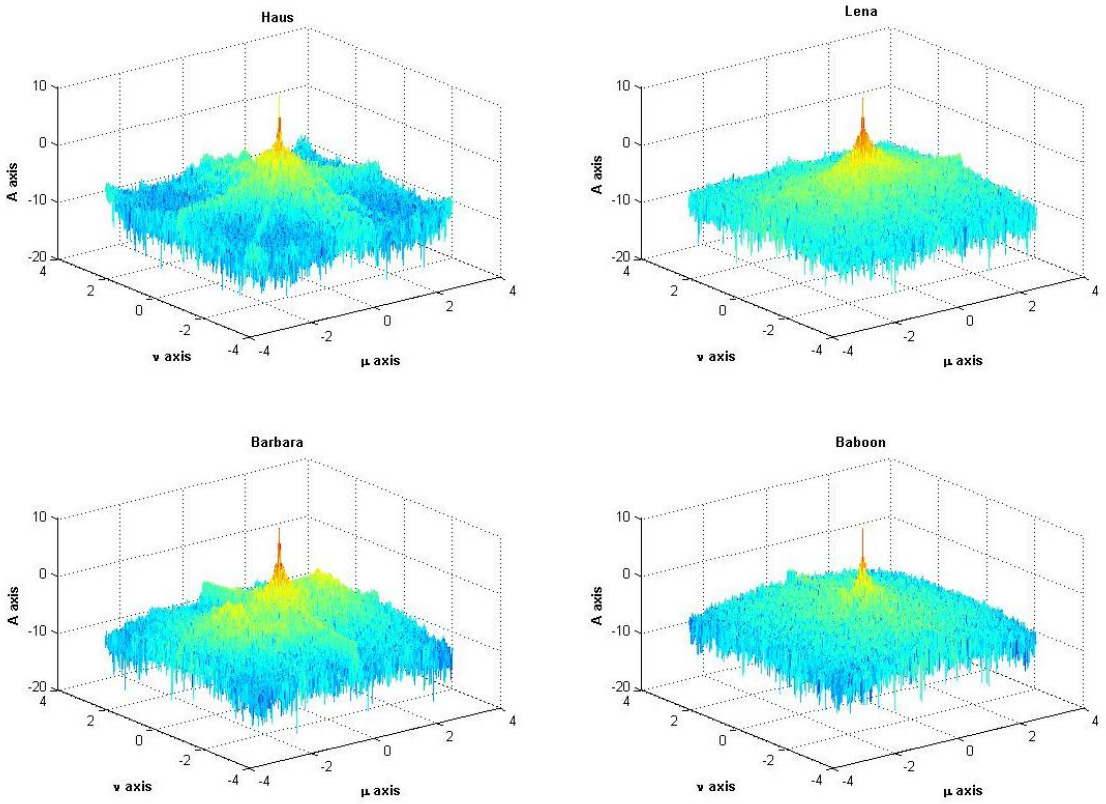


Abb. 6.2: Die logarithmischen Leistungsdichtespektren der Bilder *Haus*, *Lena*, *Barbara*, *Baboon* (von links nach rechts und von oben nach unten). Die μ-Achse und ν-Achse stellen die Frequenz entlang der waagerechten und senkrechten Richtung des Bildes dar. Auf der A-Achse sind die logarithmischen Leistungsdichtespektren der Bilder aufgetragen.

Es ist manchmal schwierig, durch die Verwendung von LDS zu zeigen, bei welchem Bild die Wirksamkeit von Codierungsverfahren am besten ist. Deswegen werden wir auch das Maß spektraler Konstanz (MSK) berechnen, um zu zeigen, dass zwar das LDS des Bildes *Haus* in diagonaler Richtung eine hohe Leistung aufweist und das Codierungsverfahren für dieses Bild besonders effizient ist. Das Bild *Baboon* hingegen wird mit niedrigerer Effizienz codiert. Das MSK gibt uns nur einen Wert, der den Verlauf des LDS beschreibt. Für ein Bild ist es gegeben durch die Gleichung

$$\gamma_x^2 = \frac{\left[\prod_{i=0}^{M-1}\prod_{j=0}^{N-1}|F(i,j)|^2\right]^{\frac{1}{MN}}}{\frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}|F(i,j)|^2} \qquad 6.2$$

Hier ist $F(i,j)$ der $(i,j) - ste$ Fourierkoeffizient des Bildes. Das MSK liegt zwischen 0 und 1. Wenn es näher an der 1 liegt, ist die Wirksamkeit von Codierungsverfahren schlecht. Liegt es näher an der 0, ist die Wirksamkeit der Codierungsverfahren insofern sehr gut. Die MSK für unsere Bilder sind in folgender Tabelle gegeben

Tab. 6.1: Maß spektraler Konstantz der Bilder

	Haus	Lena	Barbara	Baboon
γ_x^2	3.1334e-004	0.0030	0.0016	0.0091

Wir sehen, dass das Bild *Haus* das kleinste MSK im Vergleich zu den anderen Bildern hat. Deswegen erwarten wir, dass die Wirksamkeit der Quellencodierung für dieses Bild sehr effektiv ist. Das MSK der Bilder *Lena* und *Barbara* sind nahe beieinander. Das MSK des Bildes *Baboon* ist größer als das der anderen Bilder, deswegen wird es schwierig, diese Bild effektiv zu codieren.

Wir benutzen in unserer Arbeit folgendens: zufällig generierte systematische LDPC-Code (siehe Kapitel 4), um die Paritätsvektoren der Länge 2048 bits zu generieren. Dieser systematische LDPC-Code erlaubt uns eine einfache Methode, um die Generator matrix durch die Niedrigdichten-Paritätsmatrix *H* zu erstellen.

Die Länge der Quellen-Wörter ist $\big((256/2) \times 256\big)/16 = 2048$ Bits. Die Länge der Paritätsvektoren der Codes wird variiert von 32 Bits bis zu 2048 Bits in Schritten von 32 Bits. So wurden insgesamt 64 LDPC-Codes benutzt. Wir haben die Generator Matrizen erstellt, so dass jedes Bit im Paritätsvektor die Addition in modulo2 von den Bits in dem längeren Paritätsvektor ist. Das heißt, wenn wir zum Beispiel die Information mit 32 Paritätsbits anstatt mit 64 codieren wollen, brauchen wir nur jede zweite der benachbarten Bits im Paritätsvektor der Länge 64 in modulo2 zu addieren.

Wir haben unsere LDPC-Codes so erstellt, um bei der Codierung Zeit zu sparen. Anstatt die Information bei einer neuen Generatormatrix zu multiplizieren, berechnen wir den neuen Paritätsvektor, indem wir die Bits in den längeren Paritätsvektor in Modulo2 addieren.

In mancher Literatur wird dieser LDPC-Code als LDPCA (engl:LDPC Accumulator) bezeichnet; der Nachteil dieser Code ist, dass sich ihre Leistung bezüglich der Fehlerkorrektur verringert, wenn die Code-Rate sehr klein.

Die Niedrigdichten-Paritätsmatrizen *H* in unserer Arbeit sind insofern nicht optimal, als dass sie Zirkel in dem Tanner-Graphen aufweisen (siehe Abschnitt 4.5 und 4.11). Aus diesem Grund ist der Decodierungsalgorithmus nicht optimal. Wir konnten nicht alle Matrizen frei von Zirkeln erstellen, denn das ist keine einfache Aufgabe, insbesondere wenn die die Code-Rate sehr klein.

In unserer Arbeit betrachten wir den virtuellen Kanal zwischen der zu codierenden Information bei LDPC und ihrer vorhergesagten Information als binären symmetrischen Kanal (engl:BSC) mit Fehlerwahrscheinlichkeit ε. In der nächsten Tabelle zeigen wir für manche unserer Code-Raten *R* die maximale Fehlerwahrscheinlichkeit ε, bei der noch eine fehlerfreie Decodierung ausgeführt werden kann. Dieses ε haben wir nicht durch den Density-Evolution-Algorithmus (siehe Abschnitt 4.11) berechnet, weil die Berechnung der Density-Evolution nur dann optimal ist, wenn es keine Zirkel im Tanner-Graphen des Codes gibt (siehe Abschnitt 4.5 und 4.11), doch wie wir vorher bereits erwähnt haben, haben unsere Matrizen diese Zirkel. Deswegen haben wir dieses ε in unseren Expermenten berechnet.

Tab. 6.2: Grenzwerte verschiedener unsere LDPC-für den BSC.

R	0.1563	0.3125	0.4688	0.625	0.7813	0.9844
ε	0.2412	0.2876	0.3345	0.4063	0.4375	0.4917

Wir sehen, dass unsere LDPC-Code die Informationen fehlerfrei decodieren können, wenn die Fehlerwahrscheinlichkeit ε maximal 0.4917 ist.

Die Niedrigdichten-Paritätsmatrizen *H* wurden dem Encoder und dem Decoder vor dem Streaming zugänglich gemacht. In unserer Arbeit benutzen wir stets weniger als 2048 Bits (Paritätsbits), um das Wyner-Ziv-Bild zu codieren, da bei der Codierung die Menge der gesendeten Bits/Information die Entropie der Quelle nicht übersteigen sollte.

6.2 Mehode I:Verteilte Bildercodierung mit Intra-Vorhersage-Modi

In diesem Abschnitt wird eine Wyner-Ziv-Bildercodierung namens verteilte Bildercodierung mit intra Vorhersage-Modi (engl:intra prediction modes) vorgestellt, angewendet und im Detail bewertet. Die Ergebnisse in diesem Abschnitt zeigen, dass das Rate-Verzerrung-Verhältnis dieser Methode I unter der Rate-Verzerrung-Verhältnis der Intra-Codierung des H.264 bleibt, aber für eine niedrige Bit-Rate haben wir fast die gleiche Verzerrung wie bei H.264 erreicht. Wir werden sehen, dass bei dieser Methode I wir weniger Codierungskomplexität haben, als wenn wir das ganze Bild in H.264 intra codieren. Diese Komplexität ist viel geringer bei der Nutzung feiner Quantisierung in Intra-Codierung H.264.

Methode I bezweckt eine verteilte Bild Codierung zu erstellen und das maximale PSNR zu erreichen. Dafür wird zusätzlich zu den Bilddaten auch die optimale Vorhersage-Modi zum Decoder geschickt.

In Abb. 6.3 ist der Aufbau der verteilten Bildercodierung mit intra Vorhersage-Modi zu sehen (Methode I). Wir teilen das Bild in zwei Bilder auf (WZ-Bild und Schlüssel-Bild), die Größe von WZ-Bild macht die Hälfte der Gesamtgröße des Bildes aus. Wir codieren WZ-Bild in Wyner-Ziv-Codierung mittels LDPC-Encoder und das Schlüssel-Bild mittels traditioneller Intra-Codierung H.264. Der Encoder sendet an den Decoder nicht nur den Bit-Stream des Codierten Schlüssel-Bildes, sondern auch andere Informationen (mittels *Suche des besten Vorhersage-Modus* in Abb. 6.3), um dem Decoder zu helfen, die beste Prädiktion für das WZ-Bild zu treffen.

6.2.1 Der Encoder

Aufteilung:

In *Aufteilung nach dem Schachbrett-Muster* (siehe Abb. 6.3) teilen wir das Originalbild $N \times N$ in 4×4 nicht-überlappende Blöcke auf, so dass jeder Block 16 Pixel hat. Dann erstellen wir aus dem Originalbild zwei Bilder nach dem Schachbrettmuster (siehe Abb. 6.4). Das WZ-Bild wird aus den weißen Blöcken erstellt, indem jeder weiße Block einen Block nach oben verschoben wird (außer den Blöcken in der obersten Blockzeile). Das Schlüssel-Bild wird auf die gleiche Weise aus den schwarzen Blöcken erstellt, indem jeder schwarze Block einen Block nach oben verschoben wird. Somit haben das WZ-Bild und das Schlüssel-Bild jeweils die Größe von $(N/2) \times N$. Wir nehmen an, dass dieses Aufteilungsmodell auch am Decoder zur Verfügung steht.

H264 Intra-Codierung:

Wir nutzen H.264/AVC Intra-Codierung [WSBL03], um das Schlüssel-Bild zu codieren. Wir nutzen die H.264/AVC Video-Codierung im Hauptprofil ohne

Ausnutzung der zeitlichen Redundanz. Wir benutzen diesen Intra-Codierungstyp, weil es für viele Anwendungen der effizienteste Intra-Codierungsstandard ist, effizienter sogar als JPEG2000 [DT01]. Wir nutzen in unserer Arbeit die JM7.3 Referenzsoftware, diese Intra-Codierung nutzt die Intra-Korrelation im Vergleich zu anderen alten Intra-Codierungen recht effizient aus. Die H264 Intra-Codierung hat mehrere Intra-Vorhersage-Modi von Blockgröße 4×4 bis 16×16 zur Auswahl.

Die kleine Blockgröße führt zu einer signifikanten Verringerung von Block-Artefakten. Es ist klar, dass die Qualität des decodierten WZ-Bildes abhängig ist vom Quantisierungsparameter *qp* in H.264.

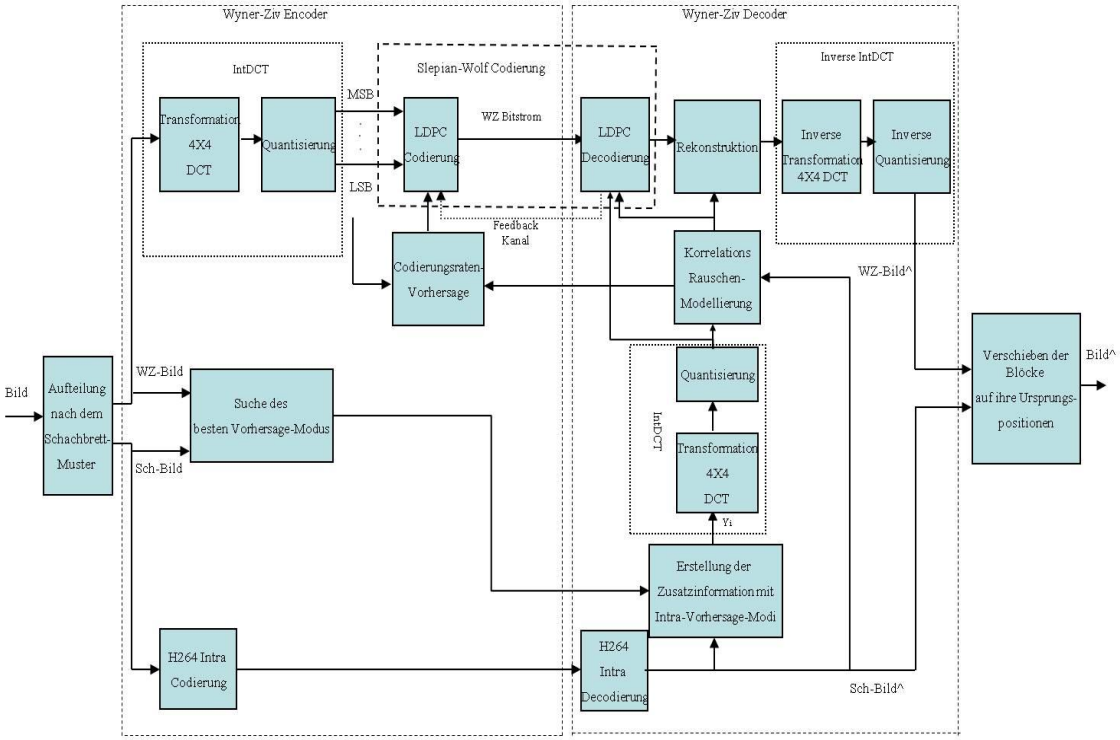


Abb. 6.3: Blockdiagramm der verteilten Bildercodierung mit Intra-Vorhersage-Modi.

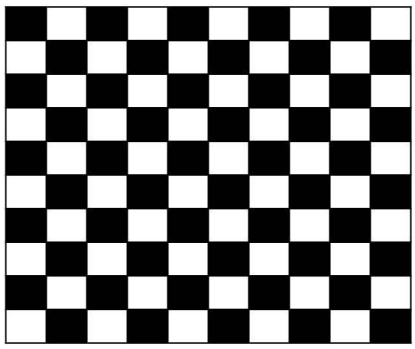


Abb. 6.4: Das Schachbrett-Muster nach dem das Bild in WZ-Bild (weiße Blöcke) und Schlüssel-Bild (schwarze Blöcke) aufgeteilt wird.

Ganzzahlige Transformation 4×4 DCT (IntDCT):

Die DCT-Transformation [KRR90] ist ähnlich der Karhunen-Loeve-Transformation (KLT), mit dem Vorteil, dass DTC-Basisfunktionen signalunabhängig sind. Diese DCT-Transformation wird bei den traditionellen Video- und Bildercodierungsstandards oft benutzt.

Der Nachteil der normalen DCT gegenüber der ganzzahligen DCT ist, dass die Elemente der Transformationsmatrix irrationale Zahlen sind. Daher erhält man nicht die gleichen Daten zurück, nachdem man eine Vorwärts- und eine inverse Transformation am Computer durchgeführt hat.

Wir wenden die ganzzahlige DCT-Transformation (IntDCT) an [MHKK03], weil alle Operationen dieser Transformation in einem 16-Bit-arithmetischen, für Input-Daten 9-Bit-arithmetischen Bereich berechnet werden können. Wir können sie in unserer Arbeit verwenden, weil der Wertbereich der Luminanzwerte 8-Bit-Daten sind. Daher können wir sagen, dass die IntDCT eine ganzzahlige Transformation (alle Operationen sind ganzzahlig arithmetisch) ohne jeglichen Verlust der Decodierungsgenauigkeit ist, und dass wir durch die ganzzahlige Arithmetik gewährleisten können, dass keine Änderungen zwischen den Ergebnissen der Vor- und Rückwärtstranformation entstehen. In der IntDCT können wir den Kern der Transformation nur durch Einsatz von Additionen und Verschiebungen (engl:shift) anwenden. Die Gesamtanzahl von Operationen ist kleiner als bei der DCT, weil ein Teil der Transformation in die Quantisierung integriert ist.

Der Kern der Vorwärts-IntDCT ist gegeben durch die Matrix

$$H = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +2 & +1 & -1 & -2 \\ +1 & -1 & -1 & +1 \\ +1 & -2 & +2 & -1 \end{bmatrix} \qquad 6.3$$

Und die inverse Matrix ist gegeben durch

$$H_{inv} = \begin{bmatrix} +1 & +1 & +1 & +1/2 \\ +1 & +1/2 & -1 & -1 \\ +1 & -1/2 & -1 & +1 \\ +1 & -1 & +1 & -1/2 \end{bmatrix} \qquad 6.4$$

Man bemerke, dass die Multiplikation bei der Nutzung der Matrix in 6.4 mit nur einer Rechtsverschiebung angewendet werden kann.

Die Gesamtanzahl von Blöcken im WZ-Bild ist gegeben durch $L = \frac{(N/2) \times N}{4 \times 4}$, wo $(N/2) \times N$ die Größe des WZ-Bildes ist, daher ist in unserer Arbeit $L = 2048$. In unserer Arbeit wenden wir eine 4×4 blockbasierte IntDCT im WZ-Encoder auf die

Luminanzwerte eines jeden Blocks im WZ-Bild an. Die Blöcke des WZ-Bildes werden in einer Raster-Scan-Reihenfolge eingescannt, von links nach rechts und von oben nach unten, so dass jeder Block eine Scanning-Ordnungsnummer hat, wobei $L-1 \geq l \geq 0$ ist. Nach Verwendung der IntDCT auf die 4×4 Blöcke werden die 16 korrelierten Samples in jedem 4×4 Block in 16 unabhängige DCT-Transformationskoeffizienten in der räumlichen Frequenzdomäne konvertiert. Diese Koeffizienten werden in 4×4 Blocks angeordnet. Wir bezeichnen diesen Block als IntDCT-Koeffizienten-Block (siehe Abb. 6 .5). Das obere linke Element in diesem Block wird als DC-Koeffizient bezeichnet, es korrespondiert mit der räumlichen Frequenz Null und stellt nichts anderes als den Mittelwert aller Sample-Werte des Blocks dar. Die verbleibenden Koeffizienten sind $AC_i, i = 1,2,\ldots,15$ korrespondierend mit der räumlichen Frequenz. AC_{15} ist die höchste räumliche Frequenz und befindet sich an der rechten unteren Ecke des IntDCT-Koeffizienten-Blocks.

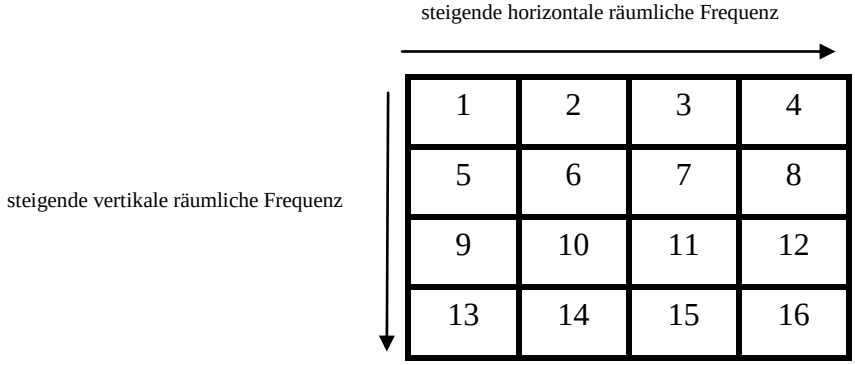


Abb. 6 .5: Der 4×4 DCT Koeffizienten-Block, die Zahlen 1:16 entsprechen den Positionen der Koeffizienten von DC bis AC15.

Die Transformationskoeffizienten in jedem der Blocks werden im Zick-Zack eingescannt, sodass jeder Transformationskoeffizient eine Scanning-Ordnungsnummer k bekommt, wobei $0 \leq k \leq (4 \times 4)^2 - 1$ ist.

Die DCT-Koeffizienten des WZ-Bildes werden anschließend nach den Positionen gruppiert, die jeder DCT-Koeffizient innerhalb unserer Blöcke eingenommen hatte. Dabei werden die Koeffizienten-Bänder gebildet. Das bedeutet, dass das erste Band aus den DC-Koeffizienten aller Blöcke und das letzte Band aus den AC15 aller Blöcke des Bildes besteht.

Quantisierung:

Die Größe des Quantisierungsschrittes QStep in H.264 nimmt mit jeder Steigerung um 6 des Quantisierungsparameters *QP* (siehe Tab. 6.3) um ca. 12% zu, so dass jede Zunahme um 6 beim Quantisierungsparameter die Größe des Quantisierungsschrittes verdoppelt. Das lässt eine breite Auswahl an Qualitätsstufen zu. In unserer Arbeit nutzen wir bei der IntDCT die beste erreichbare Quantisierungsebene. Daher arbeiten

wir stets mit *QP*=0, um die bestmögliche Qualität beim WZ-Bild, dh. mit den geringtsmöglichen Block-Artefakten, zu erzielen (Block-Artefakte würden bei schlechter Quantisierung häufiger auftreten).

Die Anzahl der Quantisierungsebenen hängt von dem Koeffizienten-Band ab. Wir haben 16 Koeffizienten-Bänder $B_c, c = 0{:}15$. Und wie bereits erwähnt haben, nutzen wir in unserer Arbeit die besten verfügbaren Quantisierungsebenen (*QP*=0) in der IntDCT. Daher haben wir immer die am besten quantisierten Koeffizienten für das WZ-Bild. Wenn *QP*=0 ist, wird *QStep*=0.6423 für die Bänder $B_c, c = 1{,}2{,}6{,}7{,}8{,}9{,}13{,}14$ und für die restlichen Bänder gleich 0.625 (für Details siehe das Buch von Richardson [Ric10]).

Wir sollten anmerken, dass diese Quantisierung über den Wertebereich des Koeffizienten-Band gleichförmig ist. Wir quantisieren jeden Koeffizienten einzeln, weil die Transformationskoeffizienten in unterschiedlichen Frequenzen im Allgemeinen als nicht miteinander korreliert betrachtet werden. Die mit AC-Koeffizienten assoziierten Basisfunktionen weisen den Mittelwert Null auf, aber die mit DC-Koeffizienten verbundenen den Mittelwert 128. Daher sind Indices von DC-Werten größer oder gleich Null, die Indices positiver AC-Koeffizienten sind größer als Null und die Indices negativer AC-Koeffizienten kleiner als Null.

Wenn *QP*=0 gesetzt wird, hat jedes quantisierte Band 11 Bits. Das führt dazu, dass der Informationsbereich für das WZ-Bild erweitert wird, da wir $(4 \times 4) \times 8$ Luminanz-Bitwerte mit 16×11 Bits wiedergeben. Bei den meisten Bildern nehmen die niedrigeren Bänder (die Niedrigfrequenz-Koeffizienten der DCT) einen größeren Bereich ein als die höheren Bänder (die Hochfrequenz-Koeffizienten der DCT). Das bedeutet in den meisten Fällen, dass die wichtigsten Bits der hohen Bänder Null sein werden. Daher nutzen wir eine kleine Anzahl von Quantisierungsebenen für die hohen Bänder und eine große Anzahl für die niedrigen Bänder.

Tab. 6.3: Quantisierung Schrittweite in H.264 codec.

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	…
QStep	0.625	0.6875	0.8125	0.875	1	1.125	1.25	1.375	1.625	1.75	2	2.25	2.5	…
QP	…	18	…	24	...	30	…	36	…	42	…	48	…	51
QStep		5		10		20		40		80		160		224

LDPC-Encoder:

Wir nutzen den LDPC-Code, um „ASWC mit der Paritätsmethode“ (siehe Abschnitt 3.3.1.2 und 4.12) auszuführen, was bedeutet, dass der LDPC-Encoder nicht die systematischen Bits des WZ-Bildes senden wird. Die Kompression wird erreicht, indem

nur die Paritätsbits des Fehlerkorrektur-Codes (LDPC) an den Decoder gesendet werden.

Jedes Band wird durch den ASWC individuell komprimiert; der LDPC kodiert den Index jedes Bandes B_c beginnend mit der signifikantesten Bitebene (MSB). Also werden die Bitebenen eine nach der anderen in den LDPC-Encoder eingespeist.

Das erste gesendete Paket in unserer Methoden enthält die Paritätsbits den DC-Koeffizienten, das zweite die AC1 und AC2 Koeffizienten, das dritte Paket AC3,AC4,AC5, das vierte AC6,AC7,AC8,AC9, das fünfte AC10,AC11,AC12,AC13 und das letzte Paket AC14,AC15. Wir haben uns für diese Vorgehensweise entschieden, weil die Niedrigfrequenz-Bänder die wichtigste Information über das Bild bereits enthalten. Daher ist es besser, wenn der Decoder zuerst die Information dieser Bänder erhält.

Codierungsraten-Vorhersage:

Die Ratenkontrolle wurde in unserer Arbeit am Encoder und am Decoder durchgeführt. Daher handelt es sich in unserem Fall um eine Hybrid-Ratenkontrolle. Deren Vorteil liegt darin, dass sie die Komplexität der Slepian-Wolf-Decodierung (insbesondere der LDPC-Decodierung) reduziert, weil die Decodierung weniger Zeit in Anspruch nimmt.

Die Bestimmung der Bitrate am Encoder ist kein einfaches Problem, da am Encoder keine Zusatzinformation (das vorhergesagte WZ-Bild bei dem Decoder) zur Verfügung steht. Bei den meisten WZ-Codierungen existiert aber ein Feedback-Kanal vom Decoder zum Encoder. Durch die Ratenkontrolle der Decoder kann die Bitrate jedem Decodierungsprozess einer spezifischen Bitebene angepasst werden. Diese Anpassung benötigt die folgenden zwei Informationen:

1-Die Zusatzinformation muss dem Decoder bekannt sein.

2-Der LDPC-Decoder muss prüfen können, ob die decodierte Information (das decodierte Code-Wort) der originalen Information (dem originalen Code-Wort) entspricht.

Wir nutzen auch die *Codierungsraten-Vorhersage* (engl:Encoder rate estimation) in Abb. 6.3 um vorherzusagen, wieviele Paritätsbits der Encoder an den Decoder senden muss. Dieser Schritt wird die Anzahl der Nachfragen des Decoders reduzieren, was die Übertragungsverzögerung und auch die Decodierungskomplexität reduzieren wird (LDPC-Decoder ist ein iterativer Decoder, in unserer Methode liegt die Iterationenanzahl bei 30). Die *Codierungsraten-Vorhersage* sendet die anfängliche Anzahl an Paritätsbits zu LDPC-Encoder in Abhängigkeit von Bitebene und der Korrelationsrauschen-Modellierung (Parameters der PDF=Wahrscheinlichkeitsdichtefunktion) ab.

Die *Codierungsraten-Vorhersage* empfängt vom Decoder den Parameter der PDF. Dieser Parameter beschreibt das Korrelationsrauschen (die Entropy $H(\widehat{B_c}/B_c)$), wobei $\widehat{B_c}$ der vorhergesagte Koeffizienten-Band am Decoder und B_c der Koeffizienten-Band des Schlüssel-Bildes am Decoder ist. Die *Codierungsraten-Vorhersage* nimmt an, dass dieses Korrelationsrauschen dem Korrelationsrauschen zwischen dem original Koeffizienten-Band am Encoder und dem vorhergesagten Koeffizienten-Band am Decoder enstpricht. Basierend auf dieser Annahme kann der Encoder für jede Bitebene im Koeffizienten-Band gut vermuten, wieviel Paritätsbits gesendet werden muss (siehe Abschnitt 4.12).

Wenn für die fehlerfreie Decodierung die Anzahl der Paritätsbits zu niedrig ausfällt, wird der Decoder eine oder mehr Anfragen an den Encoder senden, was wiederum die Decodierungskomplexität erhöhen wird. Bei einer Überschätzung der *Codierungsraten-Vorhersage* wird die RD-Kurve sich im Vergleich zur reinen Decoder-Ratenkontrolle verschlechtern.

Es ist klar, dass wenn das Korrelationsrauschen korrekt modelliert wurde, wir vorhersagen können, wieviele Paritätsbits wir senden müssen. Das Problem besteht darin, dass auch bei einem korrekten Korrelationsrauschen-Modell es schwierig ist vorherzusagen, wie viele Paritätsbits wir für eine bestimmte Bitebene senden müssen. Durch dieses Modell kann der Encoder gut vorhersagen, wieviele Paritätsbits für MSBs gesendet werden müssen, jedoch oft nicht wieviele für die LSBs. Der Grund hierfür ist, dass die vorhergesagten MSBs oft mit den originalen MSBs übereinstimmen, aber die vorhergesagten LSBs oft nicht mit den originalen LSBs übereinstimmen. In unserer Arbeit senden wir deswegen mehr Paritätsbits als in der *Codierungsraten-Vorhersage* vorhergesagt, besonders für die LSBs.

Suche des besten Vorhersage-Modus:

Wir wissen, dass in der H.264 Intra-Codierung 4×4 und 16×16 Luminanz-Vorhersage-Modi verwendet werden. Der 4×4 Luminanz-Vorhersage-Modus nutzt 9 Modi, um den besten Vorhergesagten Block für den zu codierenden Block zu finden (siehe Abb. 6.6). Dann sendet H.264 diese Modus-Nummer an den Decoder. Diese 9 Modi hängen von den Nachbarblöcke ab (die Blöcke links, oben, links oben und rechts oben vom betrachteten Block), die zuvor codiert und rekonstruiert worden waren und sowohl dem Encoder als auch dem Decoder als Vorhersage-Referenz zur Verfügung stehen.

In Abb. 6.6 sehen wir diese 4×4 Luminanz-Vorhersage-Modi. In unserer Arbeit haben wir andere mit den 4 Blöcke (die Blöcke unten, oben, links und rechts vom codierten Block) arbeitende Modi konstruiert. Unsere neuen Vorhersage-Modi wandten die gleichen Methoden an (4×4 Luminanzvorhersage-Modi in H.264), jedoch mit Blöcken, die sich in anderen Positionen zum codierten Block befanden. Wir arbeiten mit diesen Blöcken, weil sie Teil des Schlüssel-Bildes sind und bereits vorher am

Decoder rekonstruiert wurden. So nutzen wir diese Informationen, um den besten Vorhersage-Modus (im Sinne des besten PSNR zwischen dem Originalen und dem vorhergesagten Block) für die vorherzusagenden Blöcke im WZ-Bild zu finden. Wir haben in unserer Arbeit 32 4×4 Luminanz-Vorhersage-Modi konstruiert.

Der Encoder sendet für jeden Block des WZ-Bildes die Nummer des besten Vorhersage-Modus. Es müssen für jede Nummer 5 Bits gesendet werden.

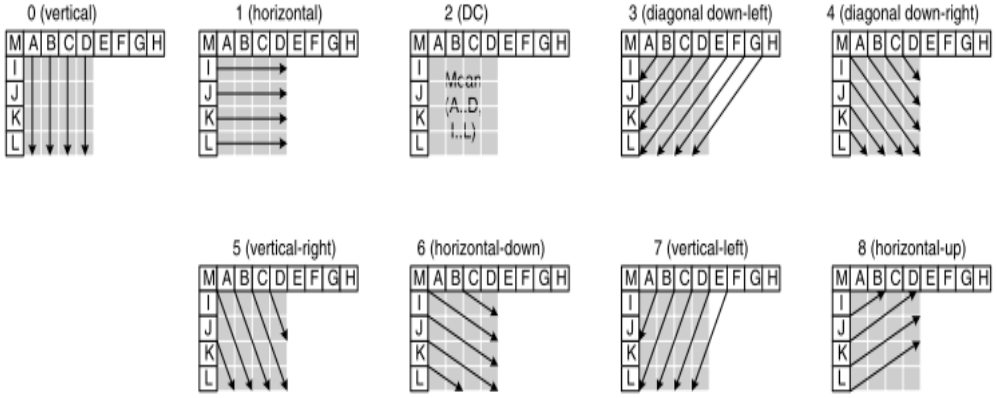


Abb. 6.6: Luminanz-Vorhersage-Modi bei H.264. die Pixel A,B,C,D gehören zum oberen Block, die Pixel I,J,K,L zum linken Block, die Pixel M zum oberen linken Block, die Pixel E,F,G,H gehören zum oberen rechten Block im Verhältnis zum betrachteten Block. Der Pfeil zeigt die Richtung der Luminanzänderung beim Vorhersage-Modus an.

6.2.2 Der Decoder

Erstellung von Zusatzinformation:

Wir gehen in unserer Methode davon aus, dass sich die Zusatzinformation (das heist das wiederhergestellte Sch-Bild am Decoder) mit dem originalen WZ-Bild deckt.

Mit Hilfe des decodierten Schlüssel-Bildes und der empfangenen Vorhersage-Modi-Nummern der Blöcke des WZ-Bildes können wir eine Vorhersage über das WZ-Bild treffen. Dieses vorhergesagte Bild ist die Zusatzinformation im Sinne der verteilten Quellencodierung. Je besser die Qualität dieser Vorhersage ist, desto weniger Fehler entstehen, die der LDPC-Decoder korrigieren muss. Diese Zusatzinformation ist faktisch eine Intra-Vorhersage des WZ-Bildes, welches codiert wurde, ohne dass die Vorhersagefehler zwischen den Blöcken und deren Vorhersagefehler gesendet wurden.

Ganzzahlige Transformation 4×4 DCT und Quantisierung (IntDCT):

Dieser Schritt am Decoder ist mit der IntDCT am Encoder identisch, was heißt, dass der Decoder den gleichen Quantisierungsschritt wie der Encoder nutzen sollte. So erhalten wir die beste Quantisierung der vorhergesagten Koeffizienten-Bänder und jedes Koeffizienten-Band hat 11 Bitebenen.

Der Decoder führt die IntDCT an der Zusatzinformation (dem vorhergesagten WZ-Bild) durch und scannt die Transformationskoeffizienten auf dem gleichen Weg ein wie der Encoder das Originalbild.

Inverse ganzzahligeTransformation 4×4 DCT und inverse Quantisierung (Inverse IntDCT):

Dieser Schritt am Decoder ist die Umkehrung der IntDCT am Encoder, was bedeutet, dass der Decoder den gleichen Quantisierungsschritt nutzen sollte wie der Encoder.

LDPC-Decoder:

Der LDPC-Decoder sollte die Fehler im vorhergesagten WZ-Bild mit Hilfe der empfangenen Paritätsbits und des Modells des virtuellen Kanals zwischen Quelle X und ihrer Zusatzinformation korrigieren. Diese Quelle ist in unserer Arbeit das Koeffizienten-Band B_c des WZ-Bildes, und die Zusatzinformation ist das vorhergesagte Koeffizienten-Band $\hat{B}_c$.

Wir gehen von einem Modell des virtuellen Kanals aus, welches eine Laplacesche Verteilung mit dem Mittelwert Null darstellt. Also gehen wir davon aus, dass die Differenz Z zwischen der Zusatzinformation und der Information am Input des LDPC-Encoders als Wahrscheinlichkeitsdichtefunktion $\frac{\alpha}{2}e^{-\alpha|Z|}$ darstellbar ist, so dass $\mathbb{E}\{Z\} = 0$ und $\sigma^2 = \mathbb{E}\{Z^2\} = \frac{2}{\alpha^2}$ sind. Diese Wahrscheinlichkeitsdichtefunktion (engl:PDF) spielt eine wichtige Rolle im Decodierungsprozess am WZ-Decoder. Der LDPC-Decoder nutzt diese PDF zur Initialisierung der Decodierungsalgorithmen, indem die Likelihood-Schätzung für die Quellenbits zur Verfügung gestellt wird. Das Parameter α der Laplaceschen Verteilung ist der "*Korrelations-Rauschen-Modellierung*" in Abb. 6.3 entnommen.

Der LDPC-Decoder könnte über den Feedback-Kanal vom Encoder mehr Paritätsbits anfordern [CWO08], andernfalls wird die LDPC-Decodierung der Bitebene als erfolgreich durchgeführt betrachtet. Der LDPC-Decoder sollte den Encoder stets informieren, ob die Decodierung erfolgreich war, damit der Encoder die nächste Bitebene aus dem Band codiert und die Paritätsbits an den Decoder sendet. Sollte der LDPC-Decoder es nicht geschafft haben, die Fehler zu korrigieren, muss er den LDPC-Encoder darüber informieren. Dieser muss dann die Bitebene mit mehr Paritätsbits codieren (das heißt, es muss ein längerer Paritätsvektor gesendet werden).

In unserer Arbeit senden wir den Bit-Wert "1" vom LDPC-Decoder an den Encoder, um diesen zu informieren, dass die Decodierung nicht erfolgreich war. War die Decodierung erfolgreich, senden wir den Bit-Wert "0". Das bedeutet, wir müssen mindestens 1 Bit für jede Bitebene über den Feedback-Kanal an den Encoder senden. Das führt bestenfalls zu einer minimalen Feedback-Kanal-Rate von $\frac{1}{(N \times N)}$ bpp (Bit pro Pixel im Bild), wobei $N \times N$ die Bildergröße ist. In unserer Arbeit, N ist 256, weshalb

wir mindestens 1.5×10^{-5} bpp brauchen. Im ungünstigsten Fall muss der LDPC-Decoder 63 Bits senden, um eine Bitebene zu korrigieren, dies entspricht $\frac{63}{(N \times N)} = \frac{63}{(256 \times 256)} \approx 0.00096$ bpp. In unserer Arbeit haben wir also festgestellt, dass die Rate des Feedback-Kanals vernachlässigbar ist.

Die Niedrigdichten-Paritätsmatrizen in unserer LDPC-Codierung sind nicht immer optimal (in dem Sinne, dass wir im Tanner-Graphen unserer Codes Zirkel haben), weil es schwer ist, einen Code zu konstruieren, der von Zirkeln frei ist, insbesondere wenn die Code-Rate klein ist. Auf Grund dieser nicht optimalen Ausgangslage braucht der LDPC-Decoder für eine erfolgreiche Decodierung viele Iterationen. Im ungünstigsten Fall muss der Feedback-Kanal öfter benutzt werden als bei der Verwendung einer Niedrigdichten-Paritätsmatrix ohne Zirkel.

Korrelationsrauschen-Modellierung:

Der Decoder berechnet das Korrelationsrauschen der Koeffizienten-Band durch das Histogramm der Differenz zwischen dem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes. Dann sendet der Decoder die Varianz σ^2 dieses Histogramms an den Encoder (*Codierungsraten-Vorhersage* in Abb. 6.3).

Der Encoder nimmt an, dass die Verteilung zwischen dem Koeffizienten-Band des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes eine Laplacesche-Verteilungskurve (Laplacesche Wahrscheinlichkeitsdichtefunktion, engl:PDF) mit variance $\sigma^2 = \frac{2}{\alpha^2}$ ist.

Dieses Laplacesche Parameter α ist für die gesamte Koeffizienten in dem Koeffizienten-Band, was heißt, dass zu jedem Koeffizienten-Band ein Laplacesches Parameter gehört. Das führt dazu, dass wir 16 Laplacesche Parameter vorhersagen müssen. Es ist wichtig, anzumerken, dass von jedem α_c, $c = 0,1, \ldots,15$ Parameter angenommen wird, es sei fest für alle Block-Positionen.

Rekonstruktion:

Der letzte Schritt in der WZ-Codierung ist der Rekonstruktionsschritt, der die Rekonstruktion des decodierten quantisierten Koeffizienten zu einem realen Transformationskoeffizienten bezweckt. In diesem Schritt nutzen wir die Zusatzinformation (das vorhergesagte WZ-Bild) zusammen mit den korrigierten Koeffizienten-Bändern, um die Rekonstruktion der Koeffizienten-Bänder des WZ-Bildes durchzuführen.

Wenn für manche Koeffizienten-Bänder keine WZ-Bits (keine Paritätsbits) übertragen werden, werden diese Koeffizienten-Bänder am Decoder durch die korrespondierenden Koeffizienten-Bänder der Zusatzinformation ersetzt. Die verbleibenden Koeffizienten-Bänder erhält man durch Anwendung einer Rekonstruktionsfunktion, die den Fehler

zwischen den Koeffizienten-Bänder des vorhergesagten WZ-Bildes und ihren korrigierten Bänder des WZ-Bildes auf die Quantisierer-Stufen beschränkt.

Diese Rekonstruktionsfunktion basiert auf der Vorhersage des minimalen mittleren quadratischen Fehlers (engl: minimum mean square error estimator =MMSE). Die Rekonstruktionsfunktion minimiert den mittleren quadratischen Fehler (engl: mean square error estimator =MSE) des rekonstruierten Wertes für jeden Koeffizienten eines gegebenen Bandes. Nehmen wir an, dass der rekonstruierte Wert am Ausgang des LDPC-Decoders q' sei und seine Zusatzinformation y. Wir wissen auch, dass der Quantisierer uniform ist und die Größe des Quantisierungsschrittes gegeben durch $\Delta = z_{i+1} - z_i, \forall i = 0,1, \dots, M-1$, wobei M die Anzahl der Quantisierungsebenen bezeichnet und $z_0 < z_1 < \cdots < z_M$ die Quantisierungsebenen selbst sind. Der rekonstruierte Wert am Ausgang des „*Rekonstruktion"* in Abb. 6.3 sollte im Bereich seiner Quantisierungsebene liegen, so dass die Initiallösung für MSE die Gleichung

$$x'_{opt} = \begin{cases} Z_i & y < Z_i \\ y & y \in [Z_i, Z_{i+1}) \\ Z_{i+1} & y \geq Z_i \end{cases} \qquad 6.5$$

Ist, wobei i den Quantisierungsindex von x und x'_{opt} den rekonstruierten Wert bezeichnet. Diese Initiallösung basiert auf dem Paper [AZG02]. In dem Fall, in dem sich die Zusatzinformation y in der rekonstruierten Teilmenge q' befindet, nimmt x'_{opt} den Wert von y an. Falls y außerhalb der Teilmenge liegt, verschiebt die Funktion den x'_{opt} in Richtung der Grenze derjenigen Teilmenge, die am nächsten zu y liegt.

Aber da der Decoder die vorhergesagte PDF zwischen X und Y kennt, können wir einen anderen Lösungsansatz [KNG07] nutzen, um x'_{opt} wie folgt zu berechnen:

MMSE sucht nach dem Wert x'_{opt}, das führt zum minimalen $MSE = E\{(x'_{opt} - x)\}$. Wir können dann x'_{opt} schreiben als

$$x'_{opt} = E[x | x \in |z_i, z_{i+1}), y] = \frac{\int_{z_i}^{z_{i+1}} x f_{X|y}(x) dx}{\int_{z_i}^{z_{i+1}} f_{X|y}(x) dx} \qquad 6.6$$

Bei $f_{X|y}(x)$ handelt es sich um die bedingte Wahrscheinlichkeitsdichtefunktion von X bei gegebenem y. Der Decoder kennt das vorhergesagte Laplacesche Modell der Differenz zwischen der Quelle Koeffizienten-Band X und der Zusatzinformation Koeffizienten-Band Y, daher erkennt der Decoder das Laplacesche Modell $f_{X|y}(x) = \frac{\alpha}{2} e^{-\alpha|x-y|}$, in dem die Varianz dieser Laplaceschen Verteilung durch $\sigma^2 = \frac{2}{\alpha^2}$ gegeben ist. Für den gleichen Wert y in der Gleichung 6.5 haben wir dann

$$x'_{opt} = \begin{cases} Z_i + \frac{1}{\alpha} + \frac{\Delta}{1 - e^{\alpha\Delta}} & y < Z_i \\ y + \frac{\left(\gamma + \frac{1}{\alpha}\right) e^{-\alpha\gamma} - \left(\delta + \frac{1}{\alpha}\right) e^{-\alpha\delta}}{2 - (e^{-\alpha\gamma} + e^{-\alpha\delta})} & y \in [Z_i, Z_{i+1}) \\ Z_{i+1} - \frac{1}{\alpha} - \frac{\Delta}{1 - e^{\alpha\Delta}} & y \geq Z_i \end{cases} \quad 6.7$$

Wo $\gamma = y - Z_i, \delta = Z_{i+1} - y$.

Verschiebung der Blöcke auf ihre Ursprungspositionen:

In diesem Schritt in Abb. 6.3 bewegen wir nur die wiederhergestellten Blöcke des WZ-Bildes und des Schlüssel-Bildes auf ihre ursprünglichen Positionen im Originalbild zurück. Das bedeutet, dass dem Decoder die Schachbrett-Aufteilung des Originalbildes bekannt sein muss.

6.2.3 Bewertung der Ergebnisse

6.2.3.1 PSNR

In Abb. 6.7 zeigen wir das Originalbild (*Barbara*), das WZ-Bild und das Schlüssel-Bild. Die anderen Bilder (*Haus*, *Lena*, *Baboon*) sieht man in Abb. 6.8. In Abb. 6.9 und Abb. 6.10 sehen wir den Vergleich der PSNR zwischen der verteilten Bildercodierung mit Hilfe von Intra-Vorhersage-Modi und der Intra-Codierung H264.

Bemerkung zu unseren PSNR-Kurven:

Der Quantisierungsparameter für die ganzzahlige DCT (IntDCT) ist bei allen Kurven gleich Null. Das bedeutet, dass wir die beste Quantisierung in der IntDCT benutzen.

Der erste Punkt in jeder PSNR-Kurve bei unserer Arbeit entspricht der Vorhersage des WZ-Bildes, bevor die Korrektur der Transformationskoeffizienten vorgenommen wurde, also vor der Übermittlung der Paritätsbits vom LDPC-Encoder. Der zweite schwarze Punkt in jeder PSNR-Kurve entspricht der Korrektur der ersten Bitebene (engl:MSB) aus Paket 1 (DC-koeffizienten Paket). Der erste Punkt in jedem Paket (aber außer dem DC-koeffizienten Paket) entspricht der Korrektur der ersten Bitebene des Transformationskoeffizienten im Paket. Zum Beispiel: der erste Punkt in Paket 2 (das heißt, der erste Magnetapunkt in jeder PSNR-Kurve) entspricht der Korrektur der ersten Bitebene (engl:MSB) des Transformationskoeffizienten AC1 und AC2; der letzte Punkt in Paket 2 entspricht der Korrektur der letzten korrigierten Bitebene des Transformationskoeffizienten AC1 und AC2.

Wir sehen, dass der erste Punkt in Paket 1 (der Vorhersage des Bildes vor allen Korrekturen) für das Bild *Haus* den höchsten PSNR bei gleicher Qualität des Schlüssel-Bildes (d.h. bei gleichem qp-Wert) erreicht, gefolgt von *Lena*, *Barbara* und *Baboon*. Der Grund liegt darin, dass das Bild *Haus* das glatteste Bild ist und im Bild *Haus* somit eine hohe Korrelation zwischen benachbarten Pixeln vorliegt (die Werte benachbarter Pixel sind sich oft sehr ähnlich, siehe Abb. 6.2 und Tab. 6.1).

Wir sehen auch, dass für alle Bilder (*Haus*, *Lena*, *Barbara*, *Baboon*) die Verbesserung beim PSNR sehr klein ausfällt, wenn die Qualität des decodierten Schlüssel-Bildes sehr niedrig ist (qp-Parameter von H.264 über 35), also nicht größer als 3 dB; bei der schlechtesten Qualität des decodierten Schlüssel-Bildes (qp=50) dagegen nicht einmal über 1.7 dB. Dies liegt darin, dass die Vorhersage des WZ-Bildes am Decoder sehr ungenau ist und die LDPC-Decodierung solch eine große Vorhersagefehlern nicht korrigieren kann (siehe Tab. 6.2). Man bemerke auch, dass bei allen Bildern (*Haus*, *Lena*, *Barbara*, *Baboon*) im Bereich der niedrigen Qualität des decodierten Schlüssel-Bildes ebenfalls keine große Verbesserung stattfindet und dass diese Verbesserung bei allen Bildern ähnlich gering ausfällt. Wenn die Qualität des decodierten Schlüssel-Bildes sehr gut ist (für die meisten Bilder, wenn der qp-Parameter von H.264 kleiner ist als 25), ist die Verbesserung beim PSNR hoch und liegt zwischen 6.06 dB und 7.49 dB. Die größte Verbesserung weist das Bild *Haus* auf.

Beim Einsetzen hoher Qualität für das decodierte Schlüssel-Bild ist die Verbesserung bei Paket 5 (AC10 bis AC13) und Paket 6 (AC14 und AC15) fast die gleiche und liegt bei fast 2 dB. Wir sehen auch, dass der Anstieg der Verzerrungskurve für diese Pakete kleiner ist als die anderer Pakete. Dieser langsame Anstieg hat zwei Gründe.

1- Die ersten Bits für die Hochfrequenz-Koeffizienten sind Null, da in den meisten Bildern die Wertebereiche dieser Koeffizienten nahe bei Null liegen. Im Vergleich zum Bild *Baboon* hat das Bild *Haus* in diesen Paketen mehr Null-Bits.

Im Falle jedes Null-Bits braucht der Decoder vom Encoder wenig Paritätsbits, um den Decodierungstest zu machen und um zu überprüfen, ob der Bitwert Null ist. Und dies führt dazu, dass die Kosten (Bit pro Pixel) für diese Pakete 5 und 6 höher werden. Dies legt nahe, dass mehr entwickelte Quantisierung die Codierungsleistung für Hochfrequenz-Koeffizienten verbessern könnte. In unserer Methoden ist die minimale Anzahl der zu sendenden Paritätsbits gleich 32.

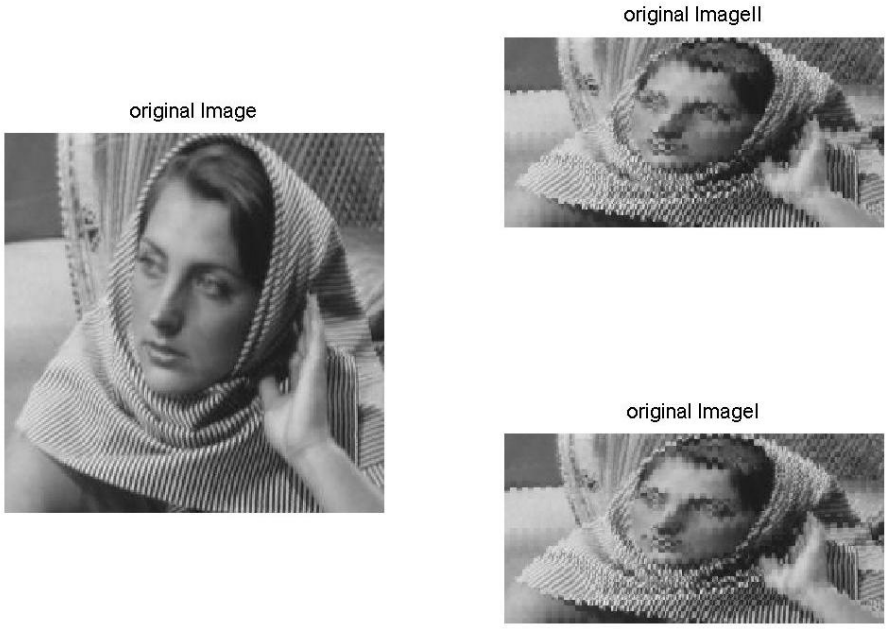


Abb. 6.7: Originalbild (*Barbara*), WZ-Bild und Schlüssel-Bild bei Schachbrett-Aufteilung.

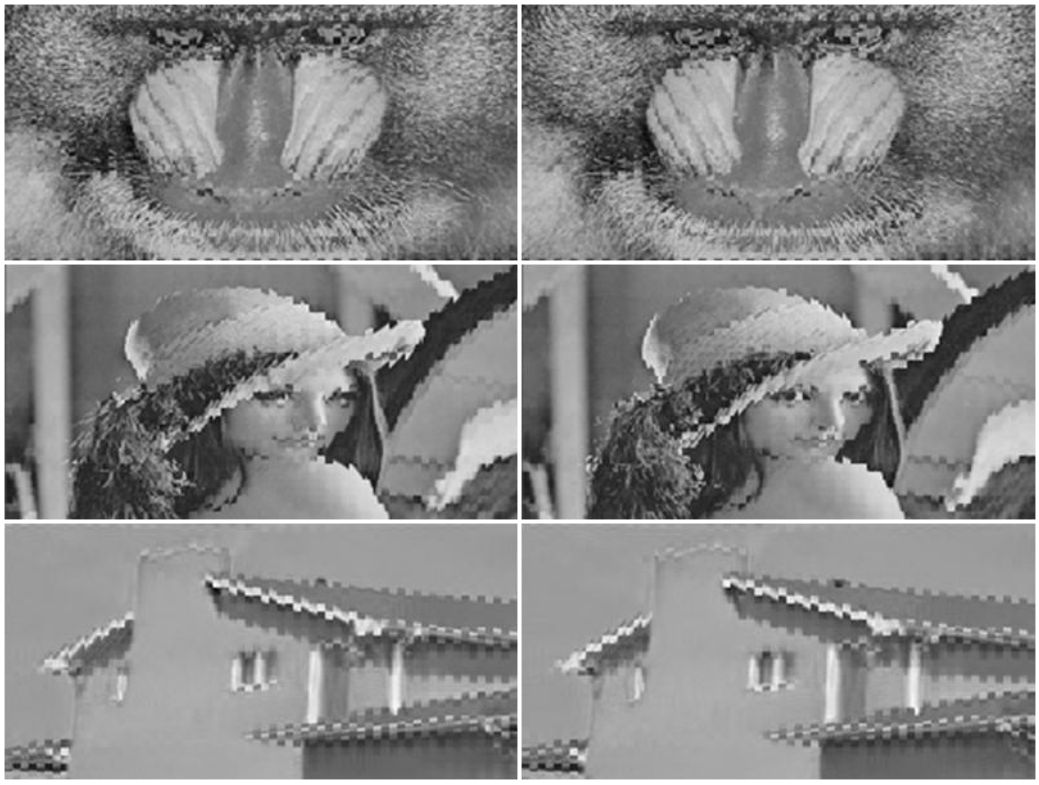


Abb. 6.8: Die Original-WZ-Bilder links und die Original-Schlüssel-Bilder rechts, die Bilder wurden mit Methode I (Verteilte Bildercodierung mit Intra-Vorhersage-Modi) codiert.

2-Die Korrektur (Verbesserung beim PSNR) wird für die Hochfrequenz-Koeffizienten nicht viel stärker sein, weil deren Energie für die meisten Bilder zu niedrig ist. Wir sehen aber, dass die Verbesserung der hohen Pakete (wie Pakete 5 und 6) im Bild *Baboon* nicht größer ist als bei den anderen Bildern, da unsere Vorhersage-Modi zwar gute Vorhersagen für die meisten Bilder treffen, aber nicht für Bilder mit starker hohe Textur, wie z.B. das Bild *Baboon*.

Wir sehen, dass für jedes Bild ab einer bestimmten Qualität des decodierten Schlüssel-Bildes bei einer weiteren Qualitätssteigerung keine Verbesserung (im Sinne des PSNR) mehr stattfindet. Diese spezifische Qualität liegt bei unseren Bildern etwa bei qp=15. Bei dieser Qualität zeigt der virtuelle Kanal zwischen der zu codierenden Bits und ihren vorhergesagten Bits keine kleinere Fehlerwahrscheinlichkeit ε in Vergleich zu einer besseren Qualität, alle unsere LDPC-Codes können diesen großen Vorhersagefehler ε nicht korrigieren (siehe Tab. 6.2). Wir sehen an Tab. 6.2 , dass unsere LDPC-Codes die Vorhersagefehler nur dann korrigieren können, wenn ε kleiner/gleich 0.4917 ist. Daher können wir diese Qualität als Obergrenze bei verteilter Bildercodierung in der Methode I definieren.

Wir sehen ebenfalls, dass die schnellste Veränderung bei der LDPC-Decodierung dann stattfindet, wenn sich für das Bild *Haus* die Qualität des decodierten Schlüssel-Bildes von qp=35 auf qp=20 ändert. Für die anderen Bilder hat die Änderung von qp=30 zu qp= [25, 20] den gleichen Effekt. So können wir sagen, dass jede kleine Verbesserung der Qualität des decodierten Schlüssel-Bildes in dem oben genannten Bereich von qp eine große Verbesserung der Fehlerkorrektur in der LDPC-Decodierung mit sich bringen kann.

Der Unterschied (im Sinne des PSNR) für alle unsere Bilder (*Haus*, *Lena*, *Barbara*, *Baboon*) zwischen der Codierung des ganzen Bildes mittels H.264 Intra-Codierung und unserer verteilten Bildercodierung ist gering, wenn die Qualität des decodierten Schlüssel-Bildes sehr niedrig ist (für die meisten Bilder bei qp über 35). Bei sehr guter Qualität fällt dieser Unterschied entsprechend größer aus (für die meisten Bilder wenn qp kleiner ist als 20). Wir sehen auch, dass der kleinste Unterschied zwischen der Methode I und H.264 Intra-Codierung für das Bild *Haus* 4.06 dB und für *Baboon* 0.61dB ist. Daraus folgt, dass zwischen der H.264 Intra-Codierung bei hoher Kompression (d.h. sehr großem qp) für nicht sehr glatte Bilder und unserer verteilten Bildercodierungsmethode keine große Leistungsdifferenz besteht. Für sehr glatte Bilder (wie das Bild *Haus*) fällt die Differenz jedoch größer aus.

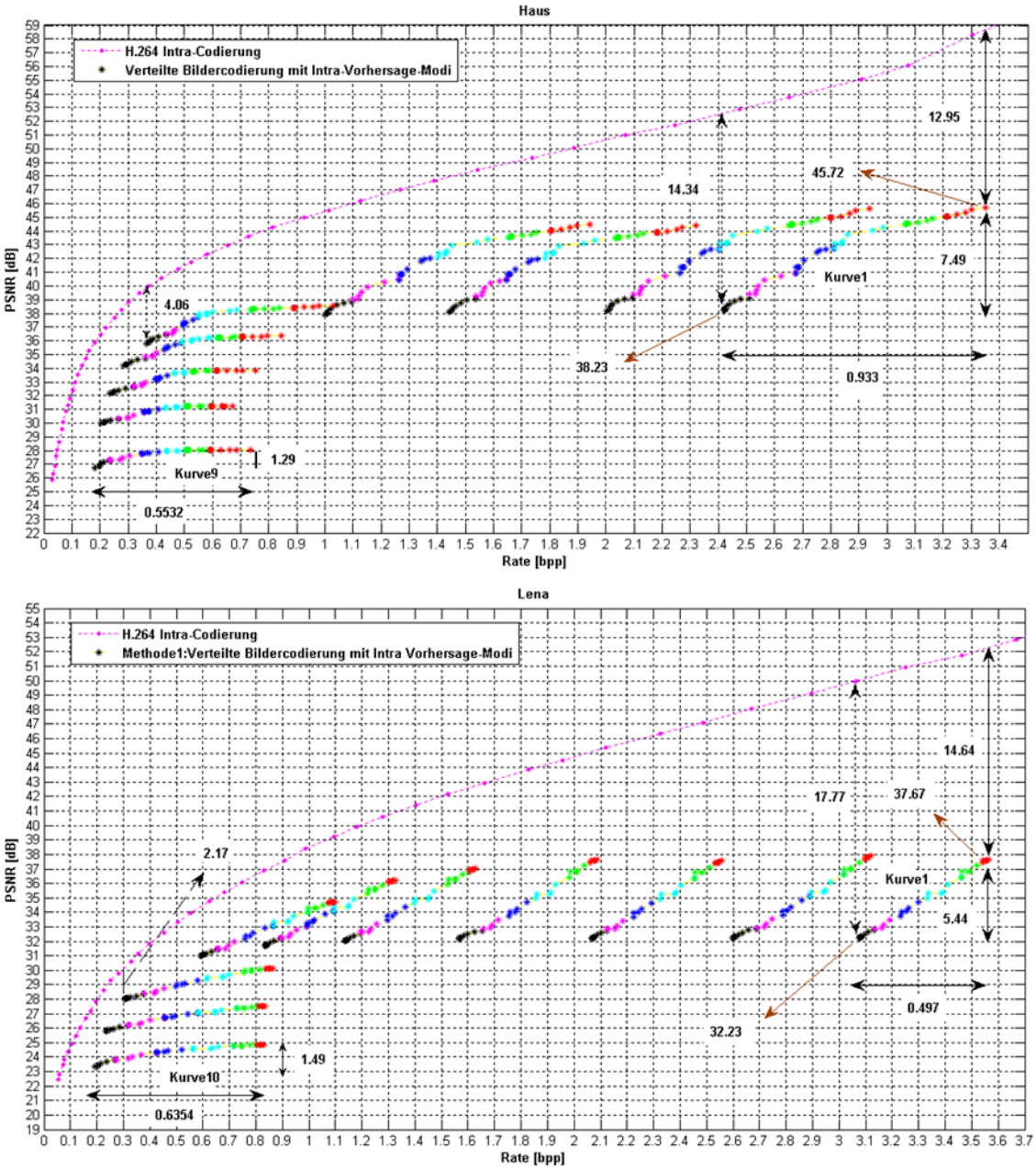


Abb. 6.9: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit Intra-Vorhersage-Modi für die Bilder *Haus* (oben) und *Lena* (unten). Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 9 sind qp=(0,5,10,15,30,35,40,45,50). Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 10 sind qp=(0,5,10,15,20,25,30,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

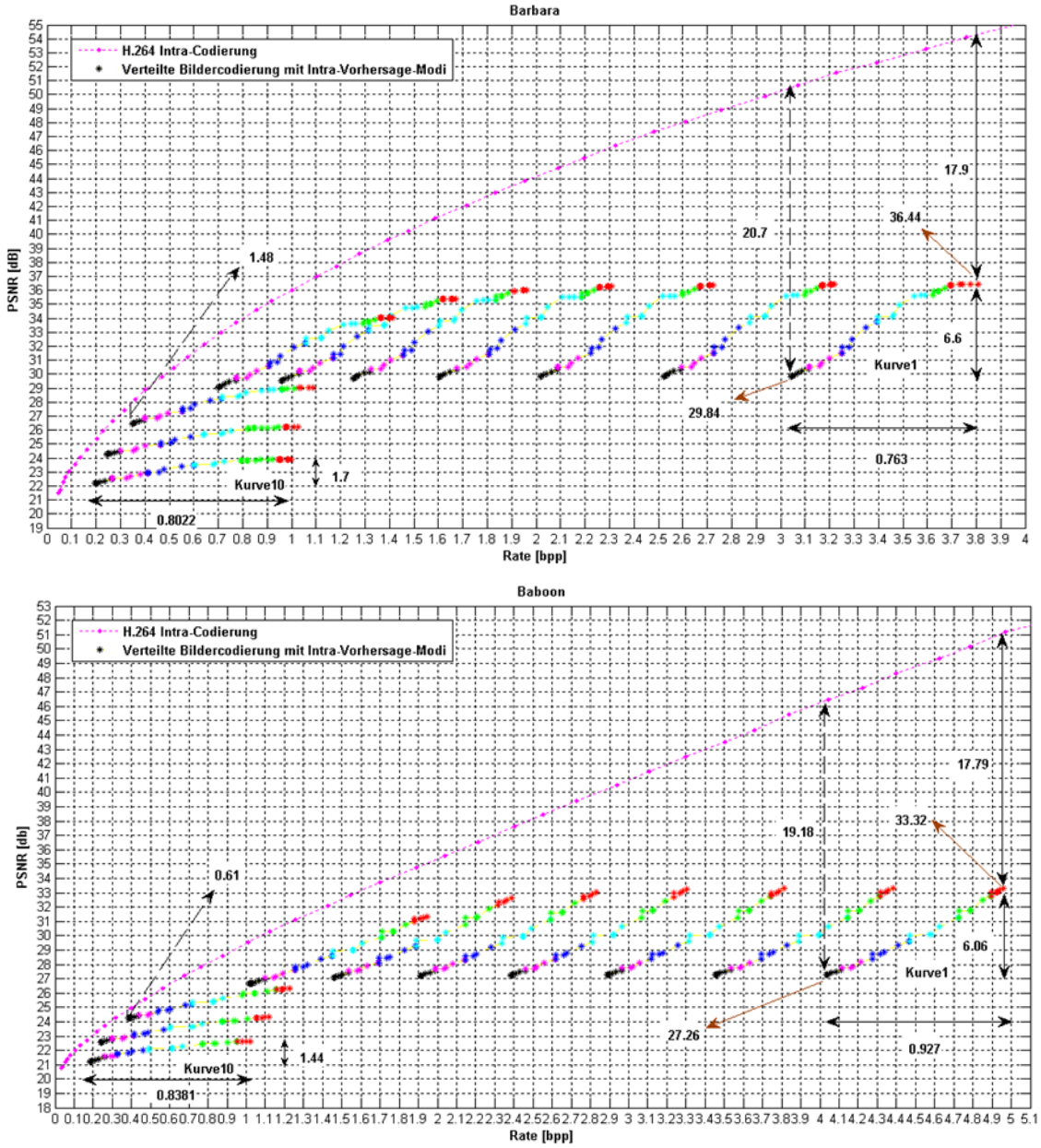


Abb. 6.10: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit Intra-Vorhersage-Modi für die Bilder *Barbara* (oben) und *Baboon* (unten). Der Quantisierungsparameter des Schlüssel-Bildes von Kurve1 bis Kurve 10 sind qp=(0,5,10,15,20,25,30,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

Wir sehen in den Abb. 6.11 und Abb. 6.12 das Maximum-optimum-Coding der Methode I im Vergleich zur Intra-Codierung H.264. Zu erkennen ist, dass bei allen Bildern der Anstieg der Optimum-coding-Kurve für hohe Bitrate klein ist (oft bei bpp größer als 0.6). Den Grund dafür haben wir oben erwähnt (die Obergrenze bei verteilter Bildercodierung in der Methode I).

Andererseits ist der Anstieg der Optimum-coding-Kurve bei niedrigen Bitraten groß: für Bild *Haus* und *Lena* im Bereich <0.4 bpp, für Bild *Barbara* für weniger als 0.5 bpp, für Bild *Baboon* im Bereich <0.6 bpp. Diesem großen Anstieg resultiert daraus, dass das vorhergesagte WZ-Bild ohne die vorherige Korrektur der Vorhersage-Fehler erreicht werden kann. Das heißt, wenn wir in diesem bpp-Bereich die Maximum-optimum-coding-Kurve erreichen wollen, sollten wir das Schlüssel-Bild in Intra-Coding H.264 besser quantisieren (durch Nutzung besserer Quantisierungsparameter QP). Wir sehen, dass bei vergrößertem bpp für das Bild *Haus* von 0.2 bpp bis 0.4 bpp das PSNR auf fast 5 dB verbessert wird; für diesen gleichen Bereich von 0.2 bpp bis 0.4 bpp wird das PSNR für das Bild *Lena* auf fast 3dB verbessert, für das Bild *Barbara* auf fast 2 dB und für das Bild *Baboon* auf fast 1dB. Das heißt, dass der Anstieg der Maximum-optimum-coding-Kurve bei diesem kleinen, erwähnten bpp-Bereich für das Bild *Haus* am höchsten wird, dagegen für das Bild *Baboon* am niedrigsten. Der Grund dafür ist, dass die Leistung der Intra-coding H.264 für ein sehr glattes Bild bei Nutzung feiner Quantizierung sehr hoch wird. Diese Leistung verschlechtert sich bei einem nicht glatten Bild (wie bei Bild *Baboon*).

Bei einer größeren Bitrate (für die meisten unsere Bilder ab 0.6 bpp) können wir die Maximum-optimum-coding-Kurve durch alleiniges Korrigieren der Vorhersagefehler erreichen, und zwar durch Nutzung der LDPC und des Feedback-Kanals.

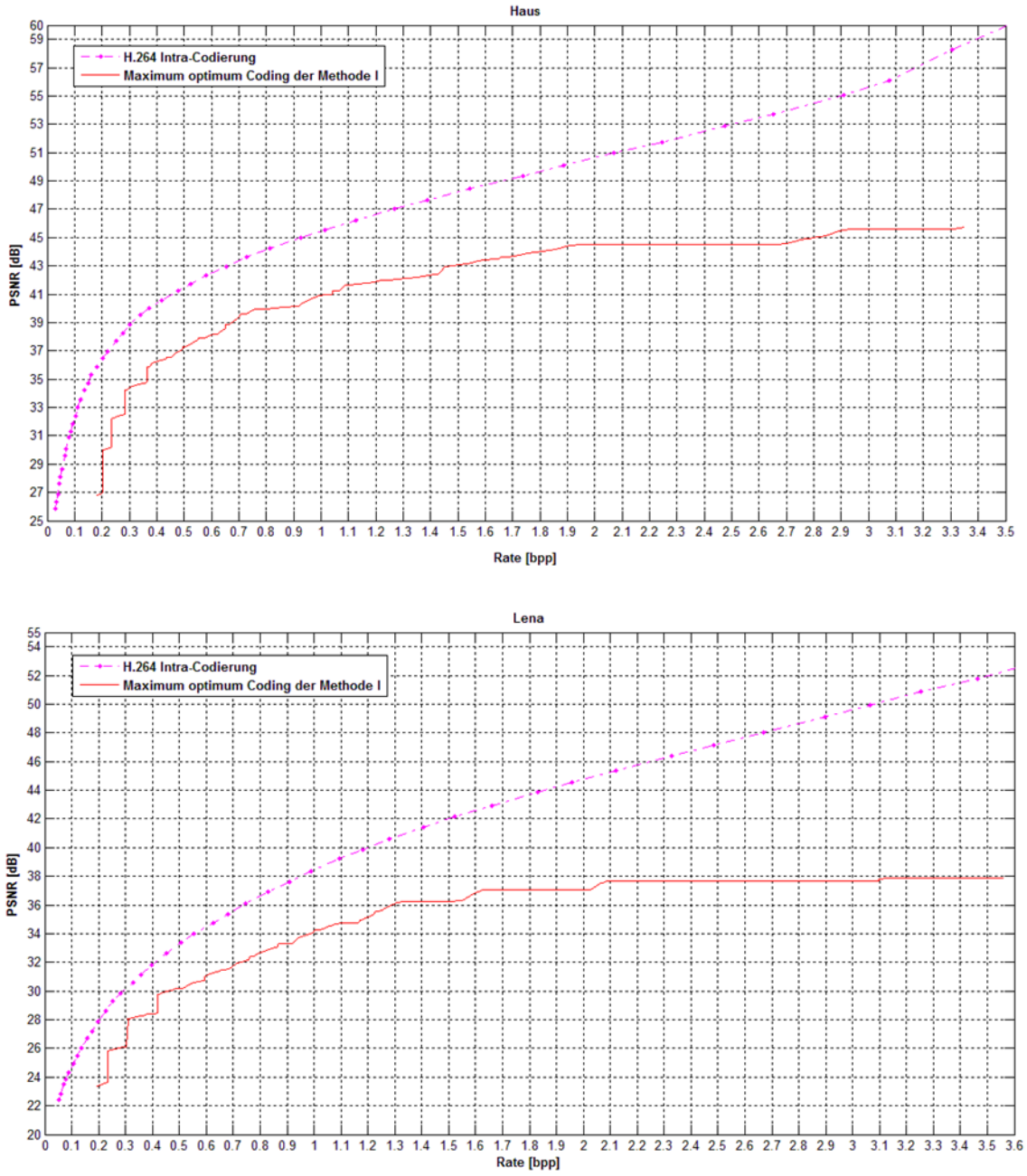


Abb. 6.11: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode I und der H.264 Intra-Codierung für die Bilder *Haus* (oben) und *Lena* (unten).

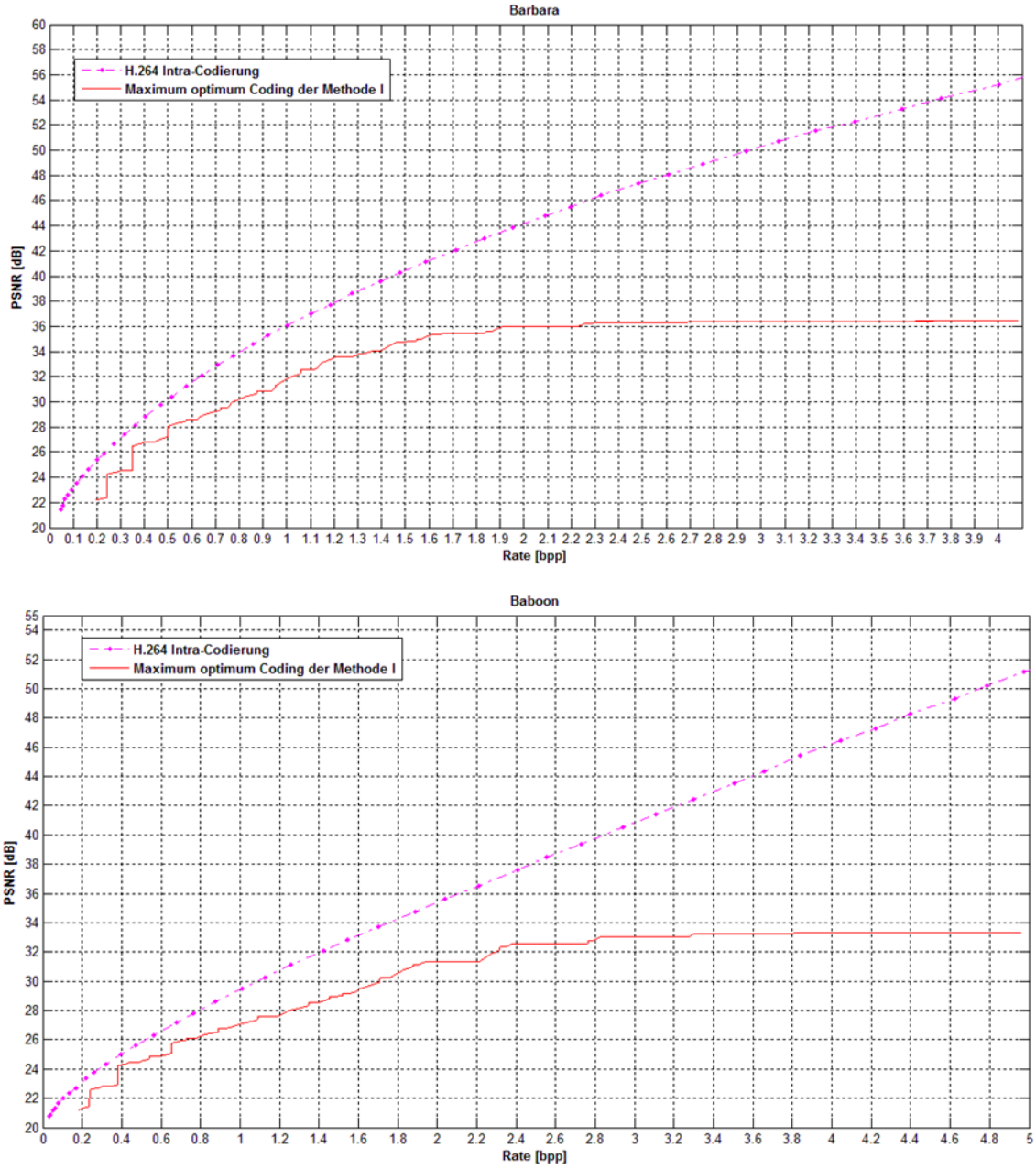


Abb. 6.12: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode I und der H.264 Intra-Codierung für die Bilder *Barbara* (oben) und *Baboon* (unten).

6.2.3.2 Kompressionsfaktor und korrigierte Bit-Rate der Bitebenen

Der Kompressionsfaktor erlaubt uns, die Fähigkeit der verteilten Bildercodierung zur Kompression der Quellendaten zu messen, wobei die Quellendaten die Bitebenen der Transformationskoeffizienten sind. Der Kompressionsfaktor CF ist gegeben durch die Gleichung

$$CF_{QP,i,j} = \frac{C_{i,j}}{P_{QP,i,j}}, wo\ QP = [0 \rightarrow 51], i = [0 \rightarrow 15], j = [1 \rightarrow 11] \quad 6.8$$

Wo QP der Qualitätsparameter bei der Codierung des Schlüssel-Bildes mit H.264 Intra-Codierung ist, ist i die Nummer der Transformationskoeffizienten, also Null für *DC* und 15 für AC_{15}, j ist die Nummer der Bitebenen (in unserer Arbeit 11). $C_{i,j}$ ist die Anzahl von Bits in einer Koeffizienten-Bitebene j eines Bandes i. In unserer Arbeit haben wir auf Grund gleichförmiger Quantisierung in der IntDCT das gleiche $C_{i,j}$ für alle i und j. Dieses $C_{i,j}$ entspricht der Anzahl von Transformationskoeffizienten in jedem Band. Daher ist $C_{i,j} = \frac{(N/2)\times N}{4\times 4} = \frac{(256/2)\times 256}{4\times 4} = 2048$, wobei $(N/2) \times N$ die Größe des Wyner-Ziv-Bildes ist. $P_{QP,i,j}$ ist die Anzahl der Paritätsbits, die für jede Bitebene j des Bandes i unter Nutzung des Schlüssel-Bild-Parameters QP gesendet werden. Also können wir in unserer Arbeit die letzte Gleichung in folgender Form schreiben:

$$CF_{QP,i,j} = \frac{2048}{P_{QP,i,j}}, wo\ QP = [0 \rightarrow 51], i = [0 \rightarrow 15], j = [1 \rightarrow 11] \quad 6.9$$

$P_{QP,i,j}$ liegt in unserer Arbeit zwischen 32 Bits und maximal 2048 Bits. Der kleinste Anstieg der Paritätsbits-Anzahl beträgt auch 32 Bits. Somit liegt der maximale Kompressionsfaktor in unserer Arbeit bei $CF_{QP,i,j} = \frac{2048}{32} = 64$ und der minimale bei $CF_{QP,i,j} = \frac{2048}{2048} = 1$. Die kleinst mögliche Veränderung des Kompressionsfaktors liegt bei 0.159. Die korrigierte Bit-Rate ist gegeben durch die Gleichung

$$CO_{QP,i,j} = \frac{K_{QP,i,j}}{C_{i,j}} = \frac{K_{QP,i,j}}{2048}, wo\ QP = [0 \rightarrow 51], i = [0 \rightarrow 15], \quad j = [1 \rightarrow 11] \quad 6.10$$

In der $K_{QP,i,j}$ die Anzahl der korrigierten Bits der Bitebene j aus dem Band i ist.

In Abb. 6.13 bis Abb. 6.16 sehen wir, dass im Allgemeinen die signifikantesten Bitebenen (engl: most significant bit=MSB, manchmal auch MSB-1) im Vergleich zu den am wenigsten signifikanten Bitebenen (engl: least significant bit-planes =LSB) eine höhere Kompression aufweisen. Das ist eine normale Beobachtung, da die Korrelation zwischen der Zusatzinformation (die vorhergesagte Bitebene) und der originalen Bitebene bei kleiner werdenden Bitebenen-Nummern zunimmt.

Man bemerke, dass $CF_{QP,i,j}$ wie zuvor gesagt nicht kleiner als 1 ist. Diese Eigenschaft ist auf den LDPC zurückzuführen, da um die Originalnachricht mittels LDPC wiederherstellten zu können, eine Code-Rate von maximal 1 ausreichend ist. Der Turbo-Code hat diese Eigenschaft nicht. Dies alles führt dazu, dass wir bei Nutzung von LDPC keine Ratenexpansion haben. Wir benötigen diese minimale Kompressionsrate, wie aus Abb. 6.13 bis Abb. 6.16 zu erkennen, um eine erfolgreiche Decodierung insbesondere bei niedriger Qualität des decodierten Schlüssel-Bildes sicher zu stellen. Bei dieser niedrigern Qulität ist die Korrelation zwischen dem Wyner-Ziv-Bild und der Zusatzinformation niedrig.

Wir sehen auch in Abb. 6.13 bis Abb. 6.16, dass die Hochfrequenz-Koeffizienten (letzte AC) üblicherweise einen sehr hohen Kompressionsfaktor für die ersten MSB_s aufweisen. Das liegt nicht daran, dass keine Vorhersagefehler für das vorhergesagte Wyner-Ziv-Bild vorliegt, sondern daran, dass der Wert dieses MSB_s für diese AC in den meisten Bildern gleich Null ist. Nur wenn das Bild wenige Intra-Korrelation zwischen den Pixeln hat, können die MSB_s dieser AC andere Werte annehmen.

Wie erwähnt wurde, werden in der Methode I für alle Koeffizienten die gleichen Quantisierungsebenen in IntDCT verwendet, und wir wissen, dass der mögliche Wertebereich der Hochfrequenz-Koeffizienten oft kleiner ist als der der Niedrigfrequenz-Koeffizienten. Daher können wir sagen, dass die Korrelation zwischen der Zusatzinformation und dem originalen WZ-Bild für die Hochfrequenz-Koeffizienten hoch ist. Wir erhalten also für die ersten Nicht-Null-MSBs der Hochfrequenz-Koeffizienten im Vergleich zu den signifikantesten Bits der Niedrigfrequenz-Koeffizienten (wie DC and AC1, AC2) höhere Kompressionsfaktoren. Wir sehen in den Abb. 6.13 bis Abb. 6.16 auch, dass für die Hochfrequenz-Koeffizienten der höchste Kompressionsfaktor für die zweite Nicht-Null-Bitebene erzielt wird (siehe zum Beispiel dass in Abb. 6.13 der Kompressionsfaktor für die Bitebene 4 in AC13 und AC14 größer ist als für die Bitebene 3). Das liegt daran, dass die Korrelation zwischen der ersten und der zweiten Nicht-Null-Bitebene dieser Koeffizienten hoch ist. In anderen Worten hat der LDPC-Decoder eine gute Vorhersage zur korrigieren (diese Vorhersage wird getroffen anhand der Wahrscheinlichkeitsdichtefunktion zwischen der Zusatzinformation und dem decodierten Schlüssel-Bild, siehe *Codierungsratenvorhersage* in Abschnitt 6.2.1) über die zweite Nicht-Null-Bitebene, wenn der LDPC die Vorinformation (engl:priori information) über die Decodierung der ersten Nicht-Null-Bitebene hat. Wenn wir eine sehr gute Qualität der Zusatzinformation haben (wenn der qp-Parameter des Schlüssel-Bildes qp 0 ist), können wir vergleichsweise mehr Bitebenen korrigieren, als wenn die Zusatzinformation eine niedrige Qualität (bei qp 50 ist) hätte. Wir sehen, dass wir bei einem sehr glatten Bild wie dem Bild *Haus* bei Anwendung einer Zusatzinformation mit hoher Qualität viel mehr Bitebenen der Koeffizienten korrigieren können (siehe in

Abb. 6.13, wir können für qp=0 für die Koeffizienten DC, AC1, AC9, AC15 mehr Bitebenen korrigieren als wenn qp=50 ist).

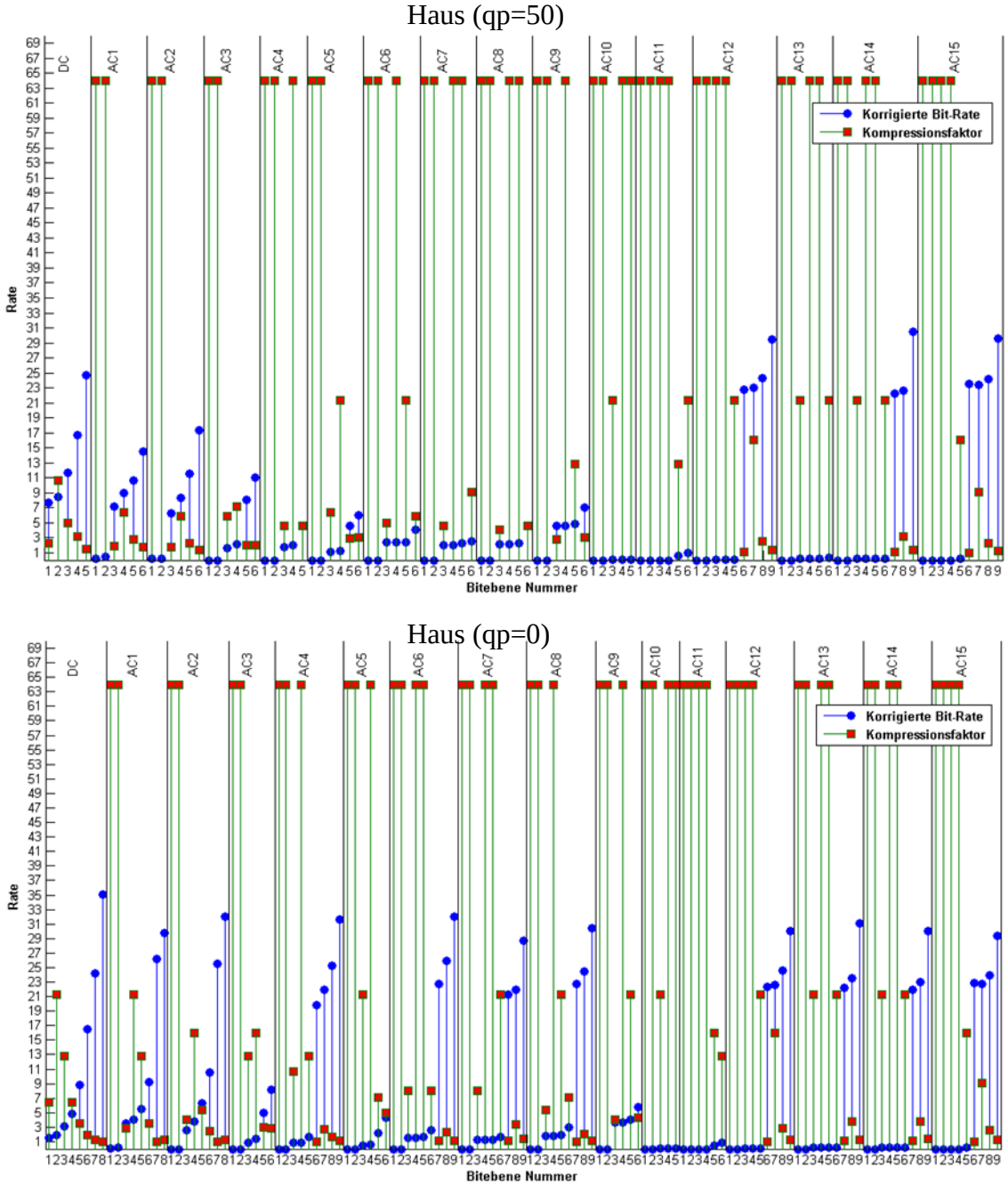


Abb. 6.13: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild Haus mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes ist qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

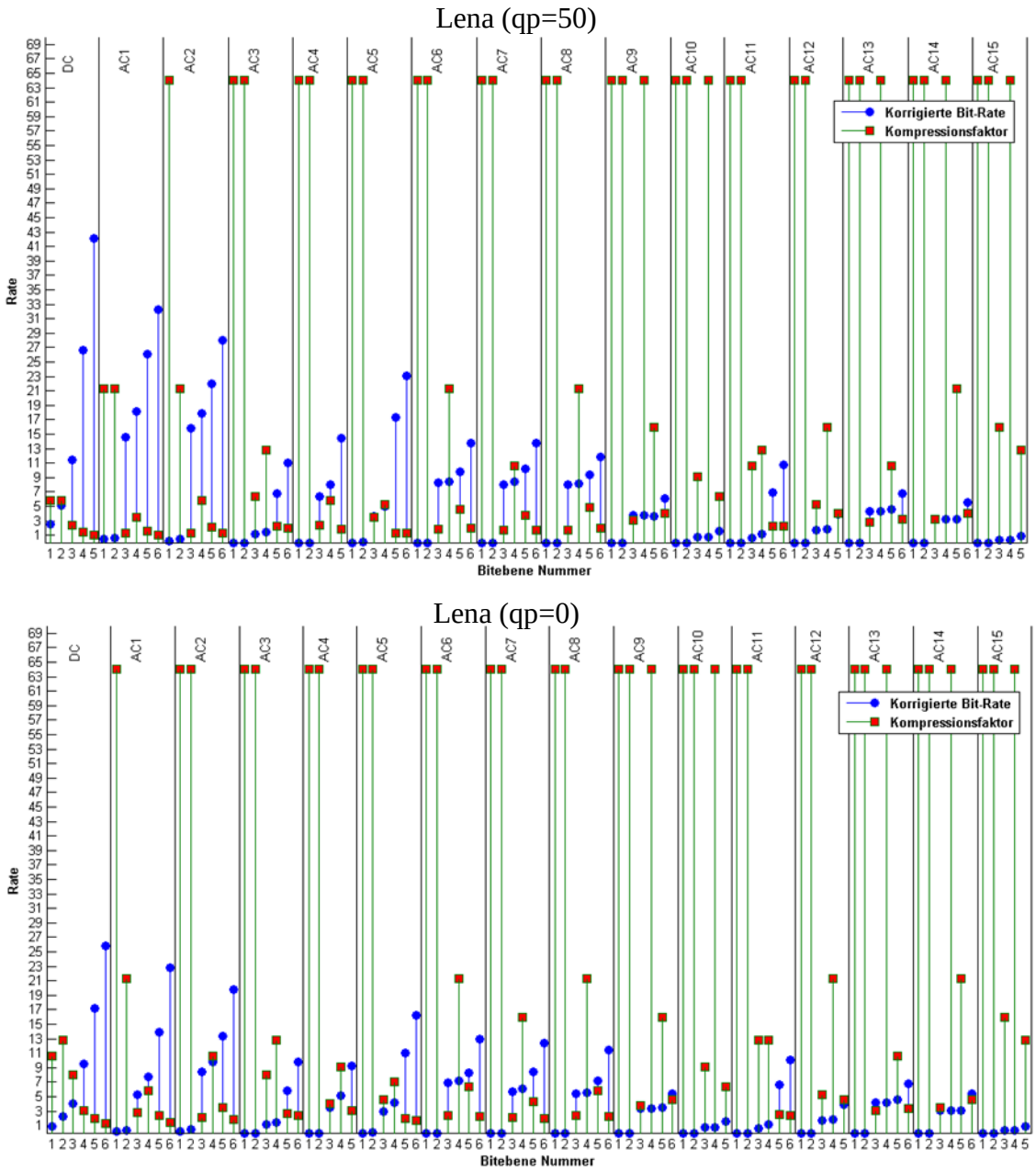


Abb. 6.14: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes ist qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

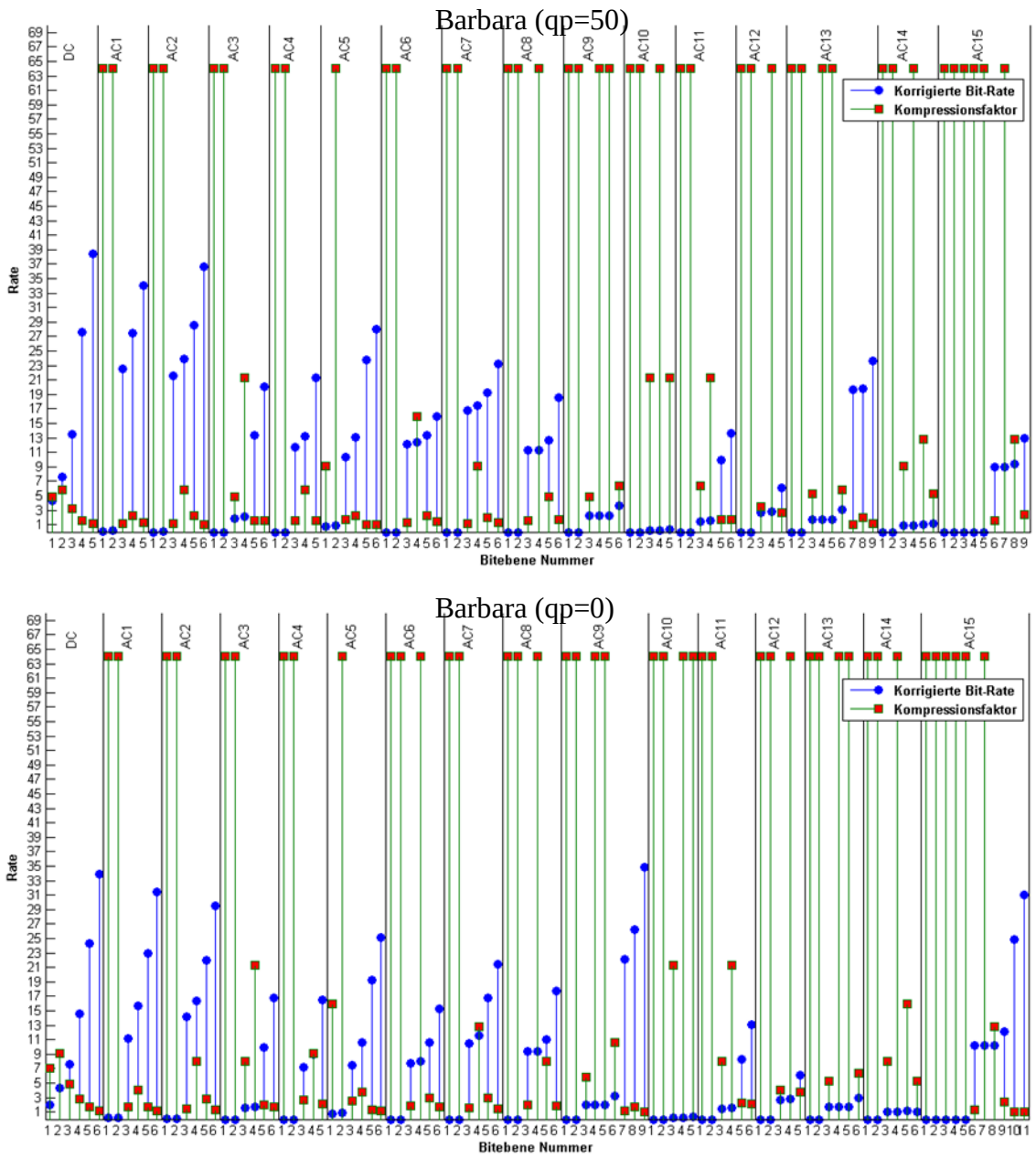


Abb. 6.15: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

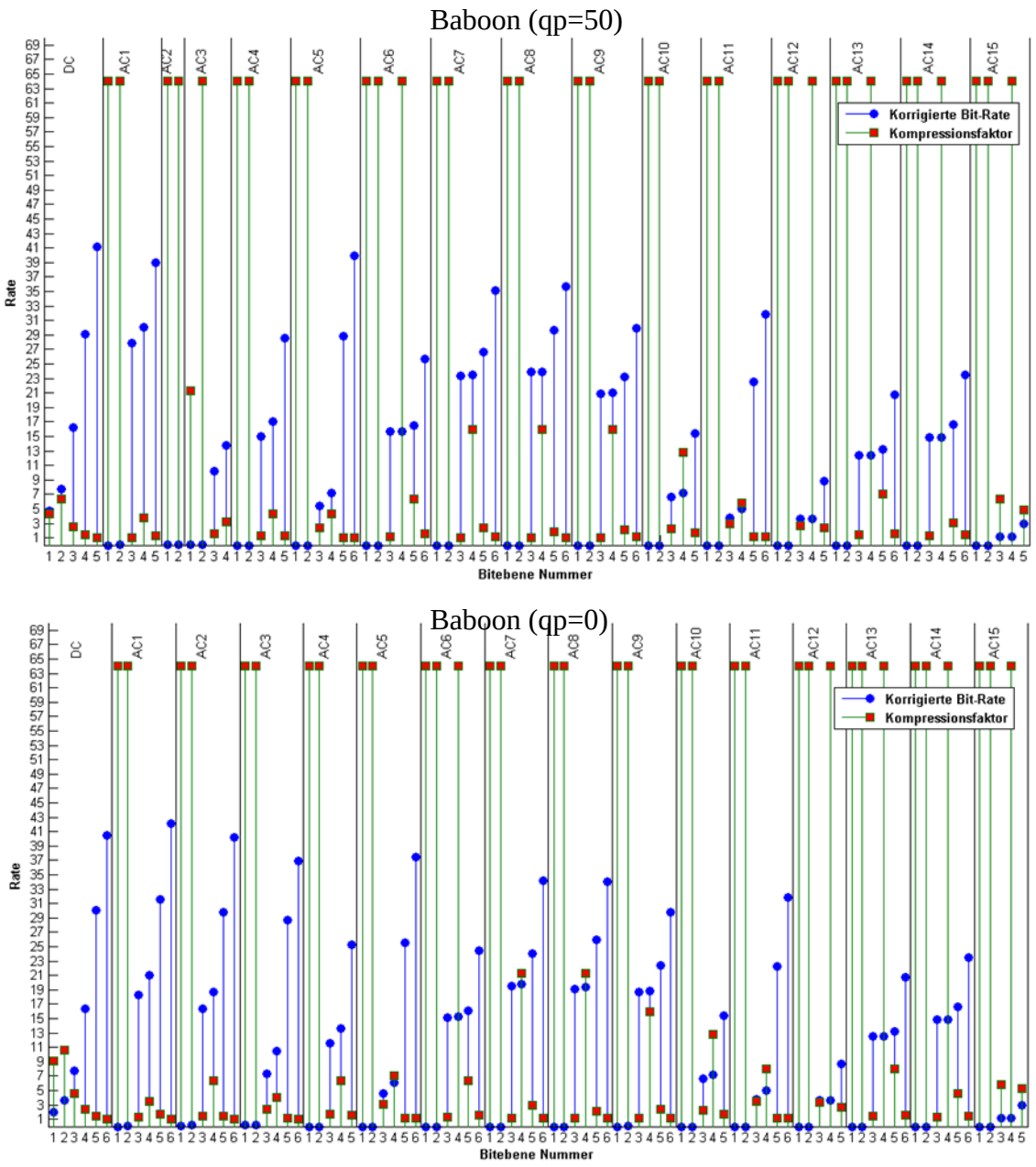


Abb. 6.16: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der verteilten Bildercodierung mit Intra-Vorhesage-Modi. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

6.2.3.3 Feedback-Kanal Rate

Der Feedback-Kanal hilft, wie worher erwähnt, dem Encoder dem kleinsten Paritätsvektoren zu finden, der für eine fehlerfreie LDPC-Decodierung ausreichend ist. Wir haben im Abschnitt 6.2.2 berechnet, dass die minimal Feedback-Kanal-Rate für die Korrektur einer Bitebene 1.5×10^{-5} bpp beträgt, und maximal 0.00096 bpp.

Bei der Methode I stellen wir fest, dass im Allgemeinen die Anzahl der durch den Feedback-Kanal zu sendenden Bits mit der Höhe der Bitebene steigt. Dies ist so, weil mit größer werdender Nummer der Bitebene die Korrelation zwischen der Zusatzinformation und dem Wyner-Ziv-Bild weniger wird. Aber die Menge der Bits, die durch diesen Kanal gesendet werden, bleibt trotzdem vernachlässigbar klein.

In der nächsten Tabelle sehen wir die Feedback-Kanal-Rate in bpp bei Anwendung der Methode I für das ganze Bild. Wir sehen, dass wenn der Quantizierungsparameter des Schlüssel-Bildes niedrig ist, die Anzahl der gesendeten Bits in dem Feedback-Kanal erhöht wird, weil unser Methode I mehr Bitebenen korrigiert, wenn die Zusatzinformation sehr gut ist (das ist der Fall, wenn die Qualität des Schlüssel-Bildes beim Decoder sehr gut ist). Außerdem sehen wir, dass das Bild *Haus* die geringste Feedback-Kanal-Rate zeigt; dagegen das Bild *Baboon* die höchste. Der Grund dafür ist, dass durch die Anwendung der Methode I bei Bild *Haus* beim Decoder ein sehr gut vorhergesagtes WZ-Bild entsteht; bei Bild *Baboon* dagegen das schlechteste.

Tab. 6.4: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode I.

Qp (Quantisierungsparameter des Schlüssel-Bildes)	Haus	Lena	Barbara	Baboon
40	0.0011	0.0013	0.0020	0.0030
35	0.0016	0.0023	0.0036	0.0052
30	0.0018	0.0025	0.0040	0.0056
25	0.0025	0.0036	0.0056	0.0074
20	0.0031	0.0051	0.0072	0.0094
15	0.0050	0.0092	0.0118	0.0145

6.2.3.4 Wahrscheinlichkeitsdichtefunktion des Korrelationsrausches der Koeffizienten

Wir haben in Abschnitt 6.2.1 (siehe *Codierungsratenvorhersage* in Abschnitt 6.2.1) erwähnt, dass der Decoder das Korrelationsrauschen der Koeffizienten durch das Histogramm der Differenz zwischendem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes berechnet. Dann sendet der Decoder die Varianz σ^2 dieses Histogramms an den Encoder (*Codierungsraten-Vorhersage* in Abb. 6.3). Der Encoder nimmt an, dass die Verteilung zwischen dem Koeffizienten-Band des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes eine Laplacesche-Verteilungskurve (Laplacesche Wahrscheinlichkeitsdichtefunktion, engl:PDF) mit variance σ^2 ist.

Wir sehen in den Abb. 6.17, Abb. 6.18 und Abb. 6.19 für das Bild *Haus* das Histogramm der Differenz zwischen dem Koeffizienten-Band (DC, AC1, AC2, AC3, AC13, AC14) des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes, außerdem zeigen wir die Laplacesche-Verteilungskurve für diese Koeffizienten-Bänder für zwei verschiedene Quantisierungsparameter (qp=0, und qp=50) des Schlüssel-Bildes. Alle diese geannten Abbildungen sind auf denselben Bereich normiert. Wir ersehen aus diesen Abbildungen wie vorher schon erwähnt und begründet, dass die Laplacesche-Verteilungskurve der Hochfrequenz-Koeffizienten enger ist als die Laplacesche-Verteilungskurve der Niedrigfrequenz-Koeffizienten. Dies führt dazu, dass mit dem LDPC eine kleinere Anzahl an Bitebenen der Niedrigfrequenz-Koeffizie zu korrigiern sind (siehe *Codierungsratenvorhersage* in Abschnitt 6.2.1), dagegen für die Hochfrequenz-Koeffizienten.

Wir sehen auch, dass wenn die Quanitisierung des Schlüssel-Bildes sehr grob ist (QP=50), die Laplacesche Verteilung nicht mit dem Histogramm übereinstimmt. In diesem Fall das Korrelationsrauschen Modell bei dem Decoder (siehe *Korrelationsrauschen-Modellierung* in Abschnitt 6.2.2) nicht gut modelliert. Aber die Übereinstimmung ist wesentlich besser, wenn die Quanitisierung des Schlüssel-Bildes sehr fein ist (QP=0).

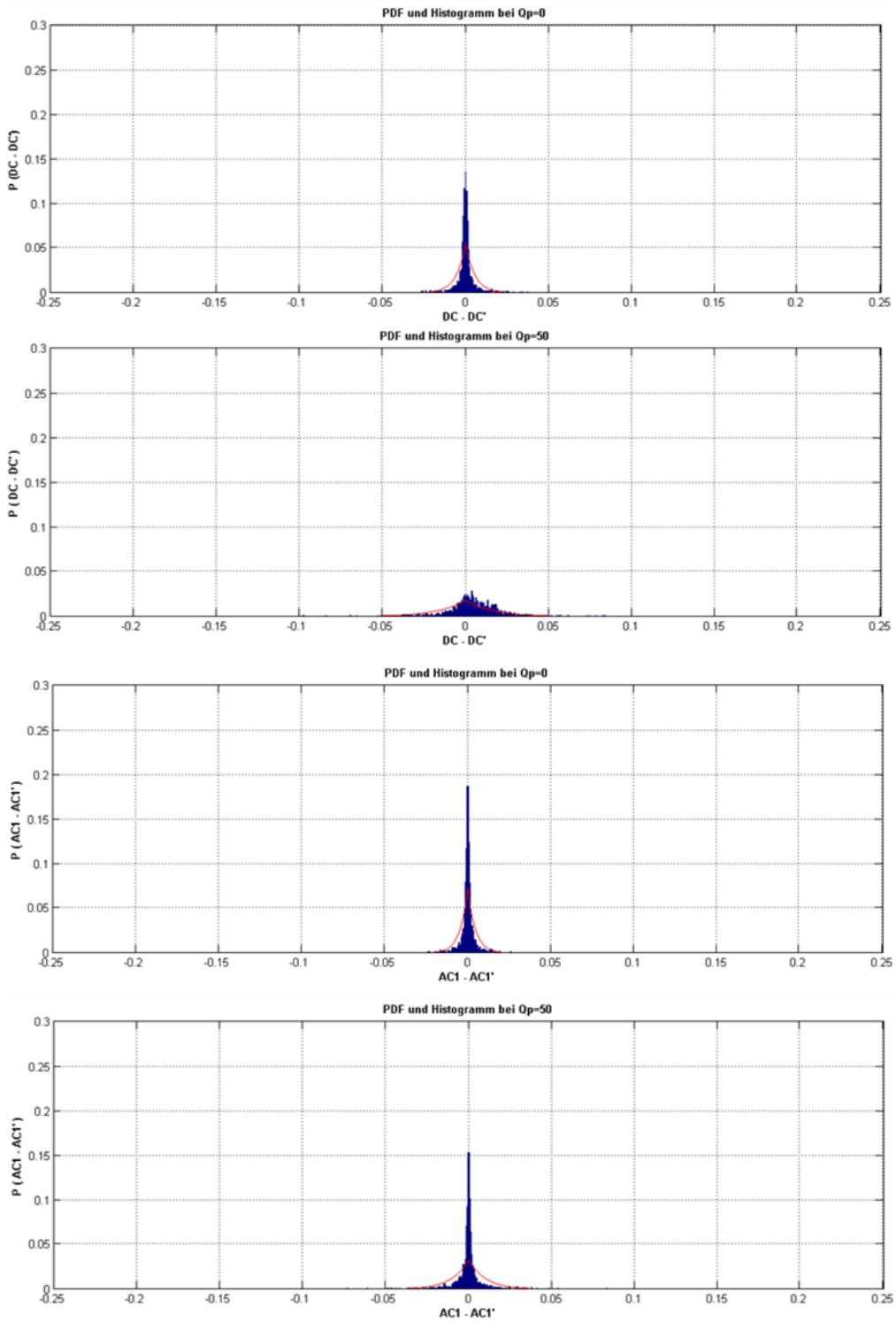


Abb. 6.17: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *DC, AC1* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet.

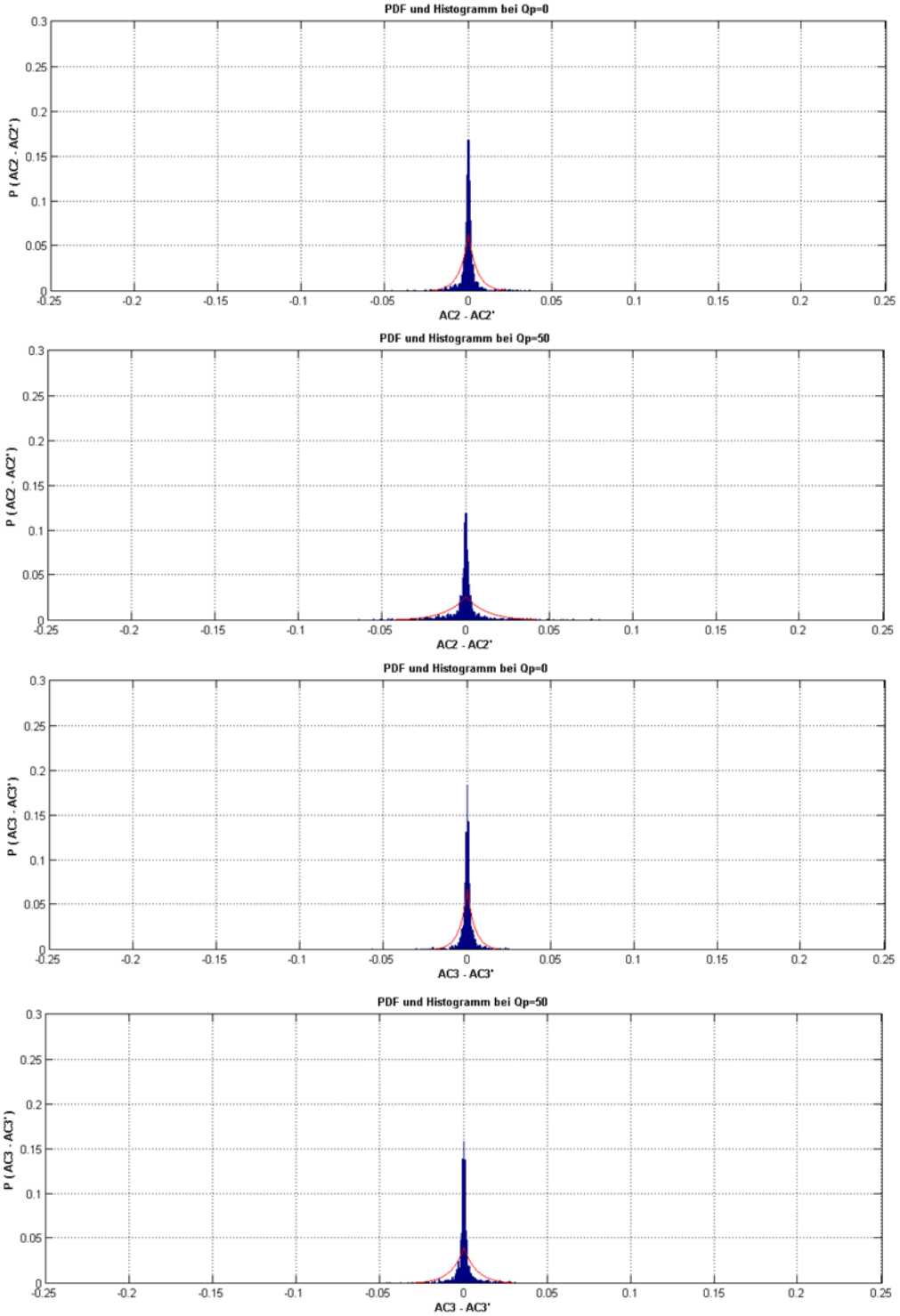


Abb. 6.18: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC2*, *AC3* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet.

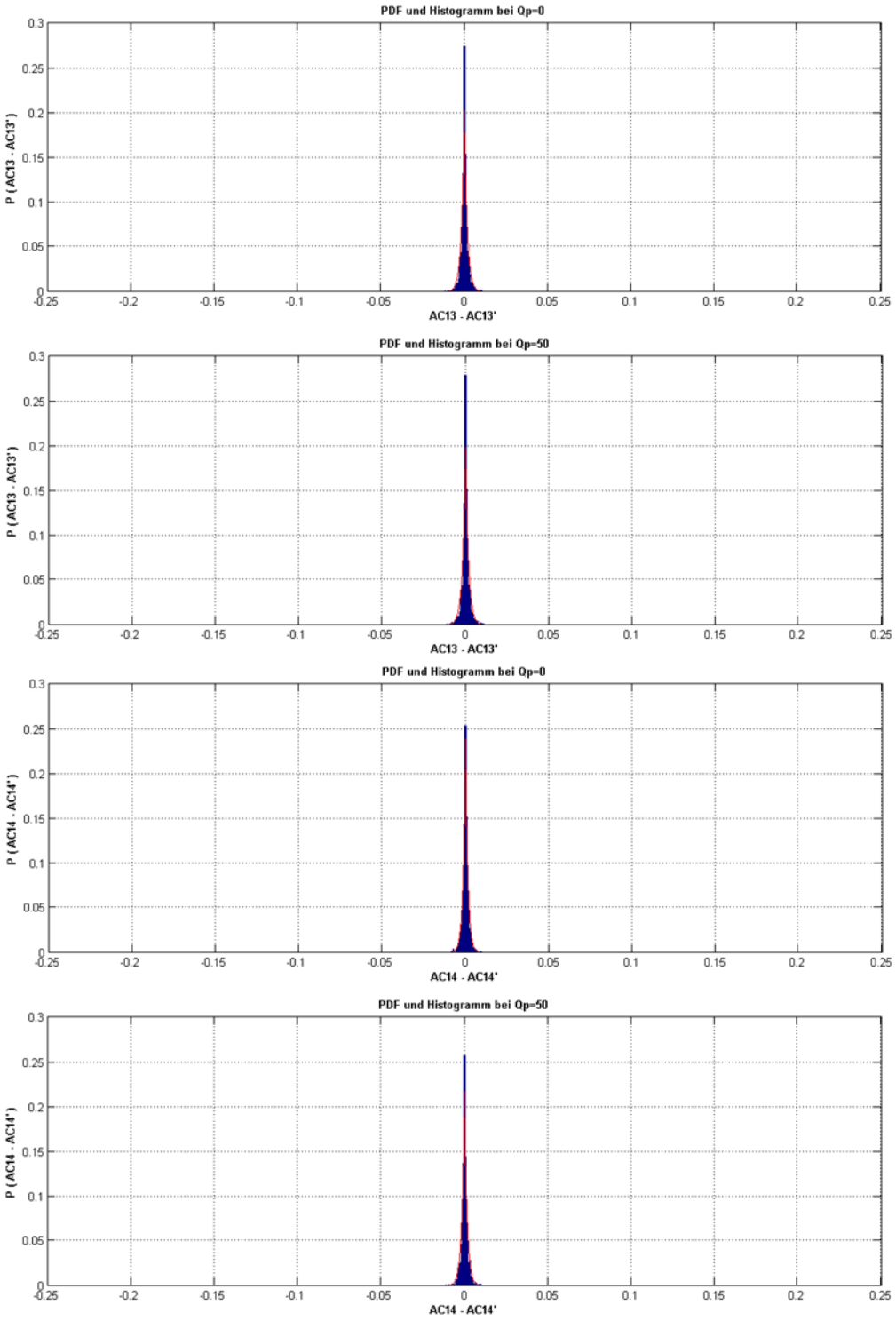


Abb. 6.19: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC13* , *AC14* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0, und Qp=50. Das Histogramm wurde für das Bild *Haus* in Methode I berechnet.

6.2.3.5 Komplexität der Encoder und der Decoder

Bemerkung: Wir werden die Komplexität der Encoder und der Decoder anhand einer Zeitmessung bestimmen. Diese Zeit hängt von der Hardware und der Software unserer Rechner ab. Als Hardware benutzen wir Intel Core 2 DUO CPU mit 3 GHZ und die *Ram* ist 3 GB. Windows XP ist das verwendete Betriebssystem. Natürlich wäre die Encoding-und die Decoding-Zeit bei einem anderen Rechner eine andere, aber das Verhältnis zwischen der Encoding-Zeit der Methode I und der Encoding-Zeit bei der Codierung des ganzen Bildes in Intra-Coding H.264 muss die gleiche sein – das Gleiche gilt auch für die Decoding-Zeit.

Komplexität der Encoder:

In Tab. 6.5 und Tab. 6.6 sehen wir den Vergleich zwischen der Encoding-Zeit der Methode I und jene, wenn wir die Bilder mit H.264 Intra-Coding codieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter des Schlüssel-Bildes für Methode I, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Die Encoding-Zeit in der Methode I hängt besonders von dem Encoding des WZ-Bildes ab. Das heißt, wir brauchen im Vergleich zu H.264 Intra-Coding weniger Zeit mit der Methode I, weil wir die Hälfte des Bildes (das WZ-Bild) mit weniger Komplexität codieren. Das LDPC-Encoding ist viel weniger komplex als das H.264-Intra Coding.

Wir ersehen aus den Tabellen auch, dass die Methode I den höchsten Quotienten ergibt (H.264 Intra-Coding/Methode I), wenn das Bild nicht sehr glatt ist (wie das Bild Baboon), weil das H.264 Intra-Coding mehr Zeit braucht, um solche Bilder zu codieren. Andrerseits braucht die Methode I nicht so viel Zeit, um solche Bilder zu kodieren, weil die Glätte des Bildes keinen großen Einfluss auf die Komplexität des LDPC-Encoding hat. Ein nicht sehr glattes Bild führt zu einem schlecht vorhergesagten WZ-Bild. In diesem Fall stimmt das Korrelationsrauschen-Modell beim Decoder nur wenig mit dem echten Korrelationsrauschen-Modell überein. Das echte Korrelationsrauschen-Modell ist das Histogramm über die Differenz zwischen den Koeffizienten-Bändern des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Ist die Übereinstimmung gering, kann die *Codierungsratenvorhersage* (siehe Abschnitt 6.2.1) nicht gut vorhersagen, wieviele Paritätsbits sie senden muss. Das bedeutet, dass bei dieser schlechten Vorhersage der LDPC-Encoder mit mehreren Anläufen Paritätsvektoren berechnen und senden muss, um eine fehlerfreie Decodierung zu erreichen; aber wir haben oben bereits erwähnt, dass die Berechnung neuer Paritätsvektoren nicht komplex ist.

Tab. 6.5: Die Zeit der Codierung in Sekunden für die Bilder *Haus*, *Lena* in Methode I im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Coding in Sekunde		Methode I in Sekunde		Quotient:H.264 Intra-Coding/Methode I	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.875	1.016	0.827	0.908	1.06	1.12
35	0.968	1.203	0.879	1.016	1.10	1.18
30	1.094	1.5	0.955	1.182	1.14	1.27
25	1.297	1.907	1.076	1.415	1.20	1.35
20	1.594	2.375	1.254	1.697	1.27	1.4
15	2.078	3	1.529	2.044	1.36	1.47

Tab. 6.6: Die Zeit der Codierung in Sekunden für die Bilder *Barbara*, *Baboon* in Methode I im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Coding in Sekunde		Methode I in Sekunde		Quotient:H.264 Intra-Coding/Methode I	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	1.125	1.157	0.982	1.014	1.15	1.14
35	1.39	1.609	1.132	1.281	1.23	1.26
30	1.734	2.203	1.338	1.643	1.30	1.34
25	2.125	2.906	1.572	2.061	1.35	1.41
20	2.531	3.609	1.797	2.487	1.41	1.45
15	3.016	4.172	2.064	2.768	1.46	1.51

Komplexität der Decoder:

Es gibt zwei Faktoren, die die Decodierungskomplexität beeinflussen. Diese sind die WZ-Bild-Decodierung und die Schlüssel-Bild-Decodierung. Wir werden auch hier wie bei der Codierungskomplexität die Decodierungskomplexität in Zeit messen.

In Tab. 6.7 und Tab. 6.8 sehen wir den Vergleich der Decodierung zwischen die Methode I und wenn wir die Bilder in H.264 Intra-Decoding decodieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter

des Schlüssel-Bildes für Methode I, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Wie ersehen aus den Tabellen, dass die Decodierungskomplexität der Methode I viel höher ist als die des H.264 Intra-Decodings. Der Grund dafür ist, dass das LDPC-Decoding viele Iterationsschritte durchführt (in unserer Arbeit benutzen wir 30 Iterations). Wir ersehen auch aus den Tabellen, dass wenn die Quantisierung des Schlüssel-Bildes sehr fein ist (wenn qp sehr niedrig ist), wird die Decodierungskomplexität höher. Das liegt daran, dass wir bei sehr guter Qualität des decodierten Schlüssel-Bildes mehr Bitebenen des WZ-Bildes korrigiern können; aber die Korrektur des LSBs benötigt längere Paritätsvektoren als die MSBs. Außerdem haben wir vorher erwähnt, dass die *Codierungsraten-Vorhersage* in Abb. 6.3 für die LSBs nicht gut vorhersagen kann, wieviele Paritätsbits er senden muss. Diese schlechte Vorhersage führt dazu, dass der Decoder mehrere Male versucht, die Decodierung mit vielen Paritätsvektoren fehlerfrei zu decodieren und jedes Mal, wenn der Paritätsvektor nicht ausreicht, vom Encoder einen längeren Paritätsvektor einfordert. Hieran wird deutlich, wie wichtig es ist, die Vorhersage des WZ-Bildes zu verbessern, denn jede kleine Verbesserung bedeutet weniger Decodierungskomplexität. Außerdem spielt die Verbesserung der Vorhersage bei der *Codierungsraten-Vorhersage* auch eine wichtige Rolle.

Tab. 6.7: Zeit der Decodierung in Sekunde für die Bilder *Haus*, *Lena* in Methode I im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

Qp	H.264 Intra-Decding in Sekunde		Methode I in Sekunde		Quotient:MethodeI/H.264 Intra-Decoding	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.047	0.062	20.91	38.86	444.9	626.8
35	0.062	0.094	36.12	86.76	582.5	923
30	0.078	0.125	45.89	115.15	588.4	921.2
25	0.110	0.187	83.14	236.41	755.9	1264.2
20	0.141	0.25	121.76	406.39	863.5	1625.6
15	0.219	0.359	242.17	874.55	1105.8	2436.1

Tab. 6.8: Zeit der Decodierung in Sekunde für die Bilder *Barbara*, *Baboon* in Methode I im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Decding in Sekunde		Methode I in Sekunde		Quotient:MethodeI/H.264 Intra-Decoding	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	0.079	0.093	80.6	157	1020.3	1688.3
35	0.109	0.156	172.1	386.3	1578.4	2476.4
30	0.156	0.25	275.2	664.6	1763.8	2658.4
25	0.219	0.344	493.9	1104.7	2255.2	3211.4
20	0.282	0.485	688.8	1668.8	2442.5	3440.8
15	0.391	0.625	1259.4	2514	3221	4022.4

6.2.3.6 Visueller Vergleich

Wir sehen in Abb. 6.20 das vorhergesagte Bild *Barbara* vor jeglichen Korrekturen (bevor Paritätsbits vom Encoder gesendet wurden), es codiert mittels verteilter Bildercodierung bei schlechter Qualität der Zusatzinformation (der Quantisierungsparameter des Schlüssel-Bildes ist qp=50), und das gleiche Bild codiert mittels H.264 Intra-Codierung bei fast gleicher Rate (bpp=0.202).

In Abb. 6.21 sehen wir das Bild *Barbara* nach der Korrektur der Bitebenen bei Anwendung der verteilten Codierung sowie bei schlechter Qualität der Zusatzinformation (qp=50), und wir sehen das ganze Bild mit H.264 Intra-Codierung, wie es bei fast der gleichen Rate (bpp=1.001) codiert. Wir sehen, dass die Korrektur sehr gering ausfällt und das Bild bei verteilter Codierung verschwommen ist. Der Grund dafür ist folgender: Wenn die Qualität der Zusatzinformation schlecht ist, nutzt der H.264-Encoder nur eine geringe Anzahl an Quantisierungsebenen, um das Schlüssel-Bild zu codieren. Dies führt dann dazu, dass die rekonstruierten Pixel jedes Blockes im decodierten Schlüssel-Bild beinahe die gleichen Pixel-Werte aufweisen. In diesem Fall erreichen die meisten Intra-Vorhersage-Modi beinahe die gleichen Pixel-Werte wie die Pixel, mittels derer die Vorhersage getroffen wurde. Die Codierung des ganzen Bildes mit H.264 Intra-Codierung führt zu einem weniger verschwommenen Bild im Vergleich zu unserer Vorhersagemethode, da H.264 den Vorhersagefehler an den Decoder sendet.

Unsere Vorhersage-Methode leidet auch an Block-Artefakten, weil wir in der Methode I die decodierten Blöcke des Schlüssel-Bildes und des WZ-Bildes auf Grund der Schachbrett-Aufteilung auf ihre ursprünglichen Positionen rearrangieren. Diese Artefakte sind besonders hoch, wenn die Qualität der Zusatzinformation sehr niedrig ist.





Abb. 6.20: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn die Qualität der Zusatzinformation sehr niedrig ist (der Quantisierungsparameter des Schlüssel-Bildes ist qp=50), PSNR dieses Bildes ist 22.18 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp.





Abb. 6.21: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn die Qualität der Zusatzinformation sehr niedrig ist (der Quantisierungsparameter des Schlüssel-Bildes ist qp=50), PSNR dieses Bildes ist 23.88 dB und die Rate 1 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.035 dB und die Rate 1.001 bpp.

Wir sehen in Abb. 6.22 das vorhergesagte Bild *Barbara* vor jeglicher Korrektur (bevor die Paritätsbits vom Encoder gesendet wurden), es codiert mittels verteilter Bildercodierung (Methode I) bei einer Qualität der Zusatzinformation, die durch den Quantisierungsparameter des Schlüssel-Bildes qp=40 gegeben ist; und das gleiche Bild codiert mittels H.264 Intra-Codierung bei fast gleicher Rate (bpp=0.362).

In Abb. 6.23 sehen wir das Bild *Barbara* nach der Korrektur der Bitebenen bei Anwendung der verteilten Codierung (Methode I), bei gleicher Qualität der Zusatzinformation (qp=40), und wir sehen auch, dass das ganze Bild mit H.264 Intra-Codierung bei fast der gleichen Rate (bpp=1.101) codiert. So sehen wir in Abb. 6.23, dass bei einer Qualitätsverbesserung der Zusatzinformation (qp=40 anstatt qp=50) bei der Methode I nicht mehr so sehr an Block-Artefakten leidet und weniger verschwommen erscheint.

Wir sehen im Anhangskapitel 8.1, dass so gut wie kein wahrnehmbarer Unterschied zwischen unserer verteilten Codierung bei sehr guter Qualität der Zusatzinformation und der H.264 Intra-Codierung besteht. Das heißt, dass die verteilte Codierung bei einer Rate von 1,66 bpp (siehe Abb. 6.11, Abb. 6.12) für viele Anwendungung ausreichend ist (es gibt keine sichtbare Differenz bei den meisten Bildern, wenn PSNR etwas über 35 dB liegt).





Abb. 6.22: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn Der Quantisierungsparameter des Schlüssel-Bildes qp=40 ist, PSNR dieses Bildes ist 26.46 dB und die Rate 0.35 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=39), PSNR diese Bildes ist 28.164 dB und die Rate 0.362 bpp.





Abb. 6.23: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi, wenn Der Quantisierungsparameter des schlüssel-Bildes qp=40 ist, PSNR dieses Bildes ist 29.04 dB und die Rate 1.09 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp.

6.2.4 Zusammenfassung

Unser Ziel in dieser Methode I war es, eine verteilte Bild Codierung zu erstellen, welche das höchtmögliche PSNR zu erreichen versucht. Wir haben die eine Hälfte des Bildes (Schlüssel-Bild) in H.264 Intra-Coding und die zweite Hälfte (WZ-Bild) mit IntDCT Transform und LDPC Encoding codiert. In der IntDCT haben wir die beste Quantisierung benutzt. LDPC-Encoding codiert den Koeffizient-Band auf Bitebene, das heißt, wir codieren zuerst die MSBs, anschließend die LSBs. Der Encoder sucht bei der Verwendung des Schlüssel-Bildes für jeden Block des WZ-Bildes den besten Vorhersage-Modus zu finden, dann sendet er die Nummer dieses Vorhersage-Modus an den Decoder. Der Decoder findet bei der Verwendung des decodierten Schlüssel-Bildes und bei der Verwendung der empfangenen Nummern der vorhergesagten-Modi das vorhergesagte WZ-Bild. Der Decoder nutzt LDPC-Decoding, um das vorhergesagte WZ-Bild zu korrigieren. Wir haben also eine Hybrid-Ratenkontrolle, um wenige Feedback-Kanal-Raten zu benötigen.

Wir haben gesehen, dass das Rate-Verzerrung-Verhältnis der Methode I unter dem Rate-Verzerrung-Verhältnis der Intra-Codierung des H.264 bleibt, aber für eine niedrige Bit-Rate (bei den meisten Bildern wenn bpp kleiner als 0.2 ist) konnten wir fast die gleiche Verzerrung wie bei H.264 erreichen, und bei dieser niedrigen genannten Bit-Rate konnten wir leider nicht so viele Bitebene korrigieren (keine Verbesserung in PSNR durch Senden eines längeren Paritätsvektoren). Wir haben auch herausgefunden, dass wir bei sehr hoher Bit-Rate die PSNR stark verbessern konnten und besteht kein großer wahrnehmbarer Unterschied zwischen der Methode I und dem H.264 Intra-

Coding. Deswegen lohnt es sich, für manche Anwendungen die Methode I statt H.264 Intra-Coding zu benutzen. Es ist bei der Methode I besonders wichtig, dass die Codierungskomplexität geringer ist als die Komplexität der H.264 Intra-Coding, aber die Decodierungskomplexität ist sehr viel höher als die des H.264 Intra-Decoding. Wir haben auch gesehen, dass die Feedback-Kanal-Rate viel geringer ist und trotzdem eine wichtige Rolle bei einer fehlerfreien LDPC-Decodierung spielt.

6.3 Methode II: Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung

In diesem Abschnitt wird eine Wyner-Ziv-Bildercodierung namens verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung vorgestellt, angewendet und im Detail bewertet. Die Ergebnisse in diesem Abschnitt zeigen, dass die Rate-Verzerrung-Verhältnis dieser Methode II unter der Rate-Verzerrung-Verhältnis der Intra-Codierung H.264 Struktur bleibt, aber für eine sehr niedrige Bit-Rate konnten wir fast die gleiche Ergebnisse der Methode1 (Abschnitt 6.2) erzielen. Wir werden sehen, dass diese Methode II besonders hilfreich ist, um nur die MSBs der Niedrigfrequenz-Transformationskoeffizienten beim WZ-Bild vorherzusagen.

Wir haben in Methode I einen bestimmten Bitstrom vom Encoder zum Decoder gesendet. Dieser Bitstrom hilft dem Decoder, die beste Vorhersage für jeden Block des WZ-Bildes zu machen. Dieser Bitstrom enthät für jeden Block des WZ-Bildes die Nummer des besten Vorhersage-Modus. Die Komplexität des Encoders, der die Vorhersage-Modi findet, ist nicht zu vernachlässigen. Deswegen werden wir bei Methode II eine andere verteilte Bild Codierung verwenden, ohne diesen Encoder zu benutzen. Wir werden bei der Methode II sogar überhaupt keinen anderen Bitstrom verwenden außer dem Bitstrom des Encoders des WZ-Bildes und dem Bitstroms des Encoders des Schlüssel-Bildes.

6.3.1 Maximal glatte Bild-Wiederherstellung mit der Transformationscodierung (MGBW)

Diese Methode II basiert auf dem Paper [WZS93]. MGBW ist eine Methode zur Rekonstruktion eines beschädigten Bildblockes aus den empfangenen Transform-Koeffizienten und der Grenz-Information (aus den Pixeln der verfügbaren benachbarten Blöcke). Diese Methode nutzt die Glätte üblicher Bilder-Signale und produziert ein möglichst glattes Bild von allen möglichen Bildern mit den gleichen empfangenen Transform-Koeffizienten und Grenz-Bedingungen.

Diese Methode minimiert die Inter-Sample-Unterschiede innerhalb des beschädigten Blocks und zwischen den angrenzenden Blöcken. Die Grenz-Information wird in die beschädigten Blocks eingeführt, so dass der Übergang an den Block-Grenzen so glatt

wie möglich erscheint. Die Methode ist besonders bei verlorengegangenen DC und einigen Niedrigfrequenz-Koeffizienten wirksam.

Das Maß für Glätte, welches in dieser Methode minimiert werden soll, ist die gewichtete quadratische Differenz zwischen angrenzenden Samples in vier Richtungen - West, Ost, Süd und Nord. Allgemein wird dieses Maß dargestellt mit der nächsten Gleichung

$$\begin{aligned}\Psi(\hat{a}_k, k \in L) = \frac{1}{2} \sum_{(m,n)\in B} \Big[& w_{m,n}^{w}\left(\hat{f}_{m,n} - \hat{f}_{m,n-1}\right)^2 \\ & + w_{m,n}^{e}\left(\hat{f}_{m,n} - \hat{f}_{m,n+1}\right)^2 + w_{m,n}^{n}\left(\hat{f}_{m,n} - \hat{f}_{m-1,n}\right)^2 \\ & + w_{m,n}^{s}\left(\hat{f}_{m,n} - \hat{f}_{m+1,n}\right)^2 \Big]\end{aligned} \qquad 6.11$$

$\hat{a}_k$ ist die Vorhersage für den verlorengegangenen Koeffizienten, B stellt einen Block dar, der aus N Samples besteht und L die maximale Anzahl der Koeffizienten (für 4×4 DCT-Transformation haben wir 16 Koeffizienten); k ist der Index des verlorenen Koeffizientent, $\hat{f}_{m,n}$ ist der vorhergesagte Wert des Sample (m, n). Die zwei Indices m und n repräsentieren die räumliche Domäne; m bezieht sich auf den Zeilen-Index und n auf den Spalten-Index.

Die Konstanten $w_{m,n}^{w},\ w_{m,n}^{e},\ w_{m,n}^{n}, w_{m,n}^{s}$ werden als Glättungsgewicht (engl: smoothing weight) bezeichnet. Sie sind von der Richtung abhängig, d.h. sie ergeben Null, wenn wir die Variationen am Punkt (m, n) in die entsprechende Richtung unterdrücken, und eins, wenn wir die Variationen in die entsprechende Richtung berücksichtigen.

Die Glättungsbedingungen (engl: smoothing constraints) sollten zwischen zwei benachbarten Samples über die Block-Grenze hinweg angewendet werden, um die Grenz-Information in den Block einzuführen. Innerhalb des Blocks sollten die Gewichte gewählt werden entsprechend der Verteilung der verlorengegangenen Transform-Koeffizienten, da die Beziehungen zwischen den Samples innerhalb des Blocks teilweise von den empfangenen Transform-Koeffizienten bestimmt werden.

Durch Minimierung der Differenz zwischen jeweils zwei benachbarten Samples in einem Block kann das Bild verschwimmen. Die Autoren zeigen auch, dass in dem Falle, dass nur der DC-Koeffizient verloren geht, die Glättungsbedingung lediglich zwischen den Grenz-Samples erzwungen werden muss (wie in Abb. 6.24 (a) gezeigt). Wenn weitere Transform-Koeffizienten fehlen, ist es notwendig, die Variation an jedem Punkt in Richtung der nächsten Grenze zu drücken (wie in Abb. 6.24 (b) gezeigt). Es wird davon ausgegangen, dass die äußeren Grenz-Samples (dunkelgrau) dem Decoder bekannt sind.

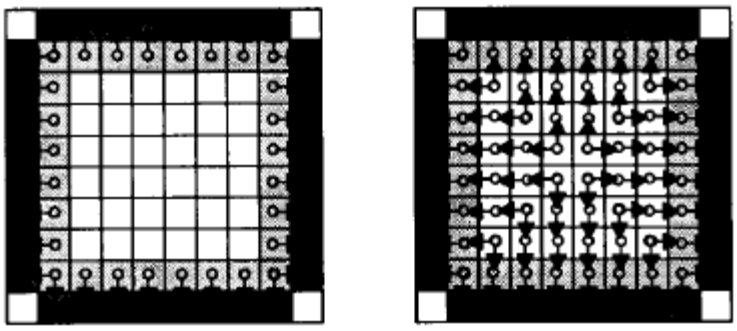


Abb. 6.24: Darstellung der Glättungsbedingungen: Ein Pfeil zwischen zwei Samples bedeutet, dass die Glätte-Differenz zwischen diesen zwei Samples im Maß für die Glätte (siehe Gleichung 6.11) vorkommt. Die dunkelgrauen Samples sind die äußeren Grenz-Samples a) Die Glättungsbedingung wird nur an der Grenze angewandt. b) Die Glättungsbedingung wird an jedem Sample in Richtung der nächsten Grenze angewandt.

Die Berechnung der verlorenen Koeffizienten (die Lösung der Gleichung 6.11) ist gegeben durch

$$\hat{a}_{opt} = (T_l^T S T_l)^{-1} T_l^T (b - S T_r \tilde{a}_r) = Ab + B\tilde{a}_r \qquad 6.12$$

$$S = S_w^T S_w + S_e^T S_e + S_n^T S_n + S_s^T S_s \qquad 6.13$$

$$b = S_w^T b_w + S_e^T b_e + S_n^T b_n + S_s^T b_s \qquad 6.14$$

Die Matrices S_w,S_e,S_n,S_s hängen von den Gewichten für die Glättungsbedingungen für die erwähnten vier Richtungen ab. Im Allgemeinen sind die Matrizen dünn besetzt. Die Vektoren b_w,b_e,b_n,b_s bestehen aus Nullen, außer an Stellen, die den Samples an einer der vier inneren Grenzen (hellgrauer Bereich in Abb. 6.24 (a)) entsprechen, sodass die Samples die Werte der benachbarten äußeren Grenz-Samples annehmen. T ist der Vektor $= [t_0, t_1, \dots, t_{N-1}]$, in dem t_k der $k$te Transformationsbasisvektor ist, bestehend aus $t_{m,n,k}$, $(m,n) \in B$. $\tilde{a}_r$ stellt den Subvektor dar, der die korrekt empfangenen Koeffizienten darstellt, und $\tilde{a}_l$ den Subvektor, der die Vorhersagen für die verlorenen Koeffizienten einschließt. T_r und T_l sind Submatrizen bestehend aus den Spalten von T, die den Eingängen von $\tilde{a}_r$ beziehungsweise $\tilde{a}_l$ entsprechen. So können wir schreiben

$$\hat{f} = T_r \tilde{a}_r + T_l \tilde{a}_l \qquad 6.15$$

In unserer Arbeit gehen wir von einer 4×4 Block IntDCT-Transformation aus, wobei die Vektoren von b die Werte der Grenz-Samples haben. Die Matrizen A und B in

Gleichung 6.12 können für verschiedene Verlustmuster im Voraus berechnet werden, und nur die linearen Transformationen, die durch diese Matrizen definiert werden, werden zur Wiederherstellung der verlorenen Informationen gebraucht. Die Dimension von A und B und somit auch die Komplexität der Rekonstruktion hängen von der Anzahl der zu rekonstruierenden Koeffizienten ab.

Indem wir $\hat{a}_{opt}$ aus Gleichung 6.12 in die Gleichung 6.15 einsetzen, erhalten wir die Lösung für den beschädigten Block.

$$\hat{f}_{opt} = T_l Ab + (T_r + T_l B)\tilde{a}_r = Cb + D\tilde{a}_r \qquad 6.16$$

Die Matrizen C und D können als Interpolations-Filter für die Vorhersage f aus den Grenzwerten b und den empfangenen Koeffizienten $\tilde{a}_r$ betrachtet werden.

In unserer Arbeit empfängt der Decoder keine Koeffizienten für die beschädigten Blöcke (Blöcke des WZ-Bildes). Zudem benutzen wir für die vier Richtungen in der Gleichung 6.11 das gleiche Gewicht $w = 1$, aber für die Blöcke an der Grenze des WZ-Bildes wird ein oder mehrere w den Null-Wert haben, weil sich in einer oder zwei Richtungen keine Samples befinden.

6.3.2 Codierung und Decodierung

In Abb. 6.25 sieht man den Aufbau dieser verteilten Codierungsmethode. Wie bei Methode I (Abschnitt 6.2) teilen wir das Bild in zwei Bilder (WZ-Bild und Schlüssel-Bild) auf. Die Größe von WZ-Bild entspricht der Hälfte des ganzen Bildes. Wir codieren das WZ-Bild mit dem LDPC-Encoder nach Wyner-Ziv und das Schlüssel-Bild mittels traditioneller H.264 Intra-Codierung. Der Encoder sendet keine Information an den Decoder, die diesem helfen könnten, die beste Vorhersage für das WZ-Bild zu finden.

Jeder Block in Abb. 6.25 ist identisch mit dem gleichen Block in Abb. 6.3. Die Vorhersage des WZ-Bildes am Decoder basiert auf der maximal glatten Bild-Wiederherstellung.

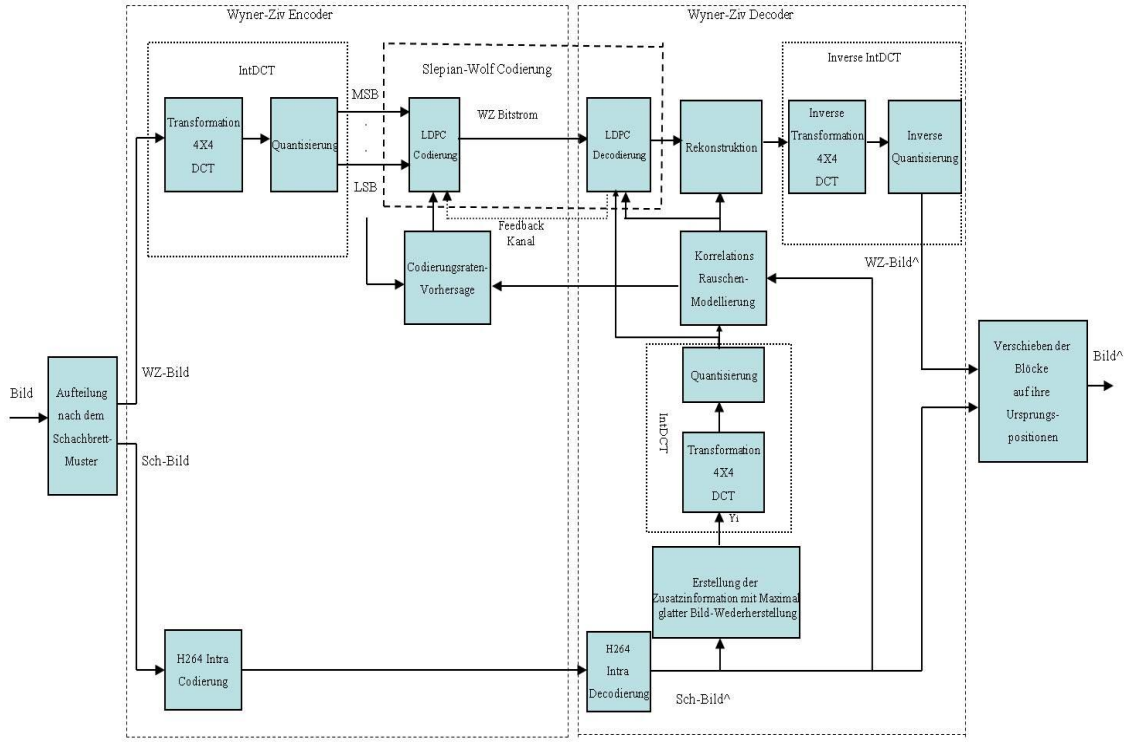


Abb. 6.25: Das Block-Diagramm der verteilten Bildercodierung mit maximal glatter Bild-Wiederherstellung.

6.3.3 Bewertung der Ergebnisse

6.3.3.1 PSNR

Abb. 6.26 bis Abb. 6.29 geben den Vergleich der PSNR zwischen der verteilten Bildercodierung mit den Intra-Vorhersage-Modi, mit der maximal glatter Bild-Wiederherstellung und der Codierung mit H.264 Intra-Codierung wieder.

Wir sehen (wie bei der Betrachtung der Methode I in Abschnitt 6.2), dass das vorhergesagte Bild *Haus* im Vergleich mit den anderen Bildern das größte PSNR erreicht. Dieser Vergleich funktioniert nur bei Anwendung des gleichen Quantisierungsparameters qp des Schlüssel-Bildes. Es gilt zu beachten, dass das vorhergesagte Bild dem ersten schwarzen Punkt jeder PSNR-Kurve entspricht.

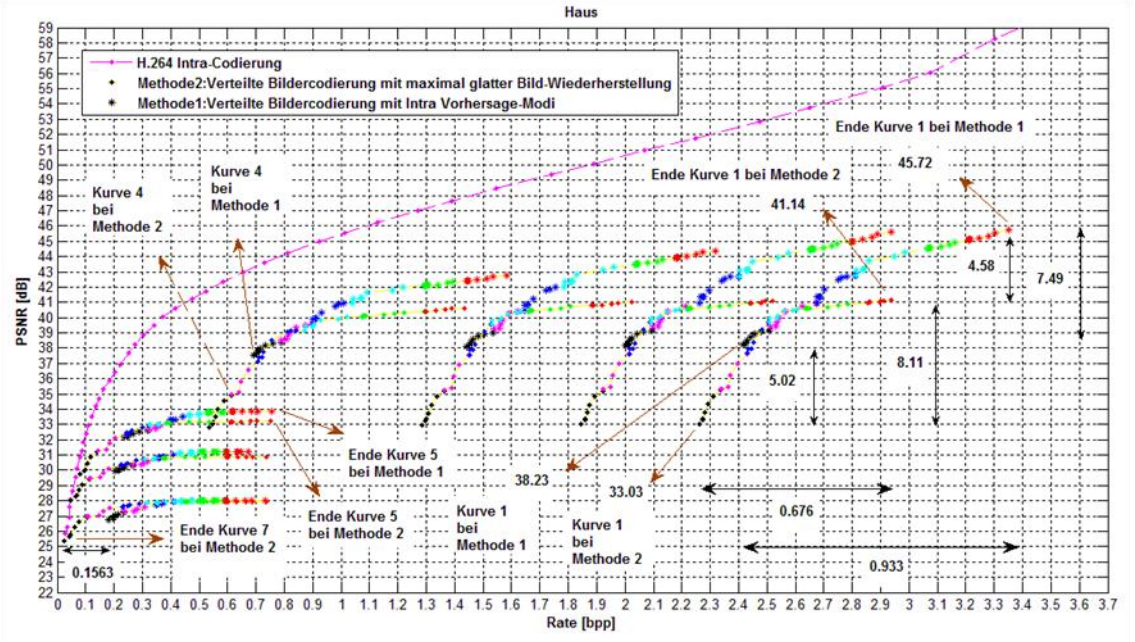


Abb. 6.26: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Haus*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 7 sind qp=(0,5,10,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

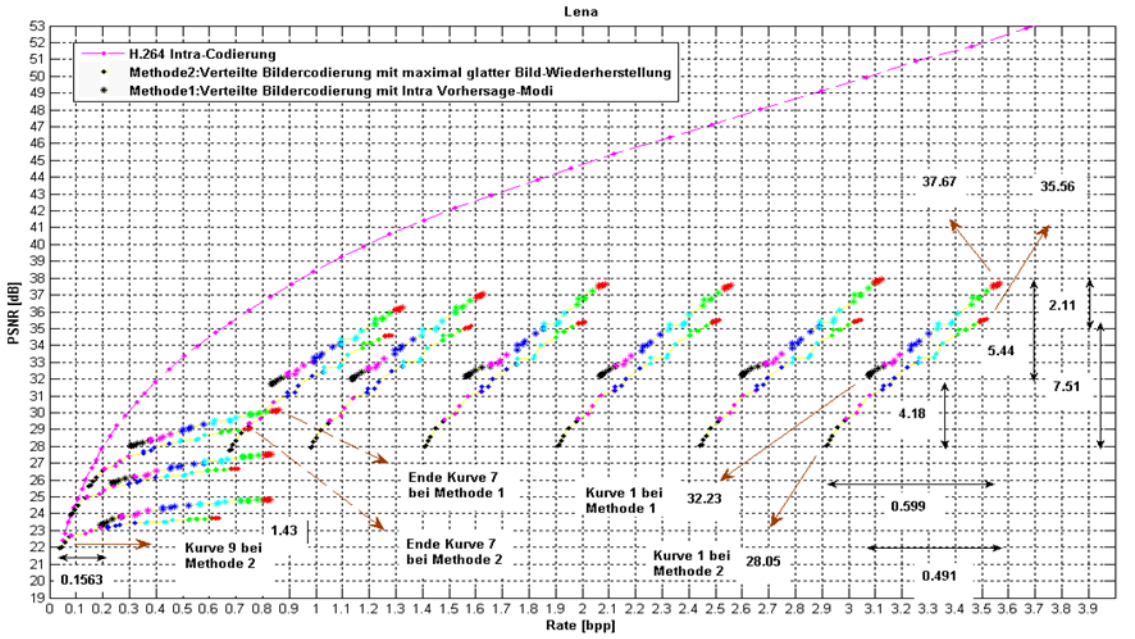


Abb. 6.27: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung,mit Intra-Vorhersage-Modi für Bild *Lena*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 9 sind qp=(0,5,10,15,20,25,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

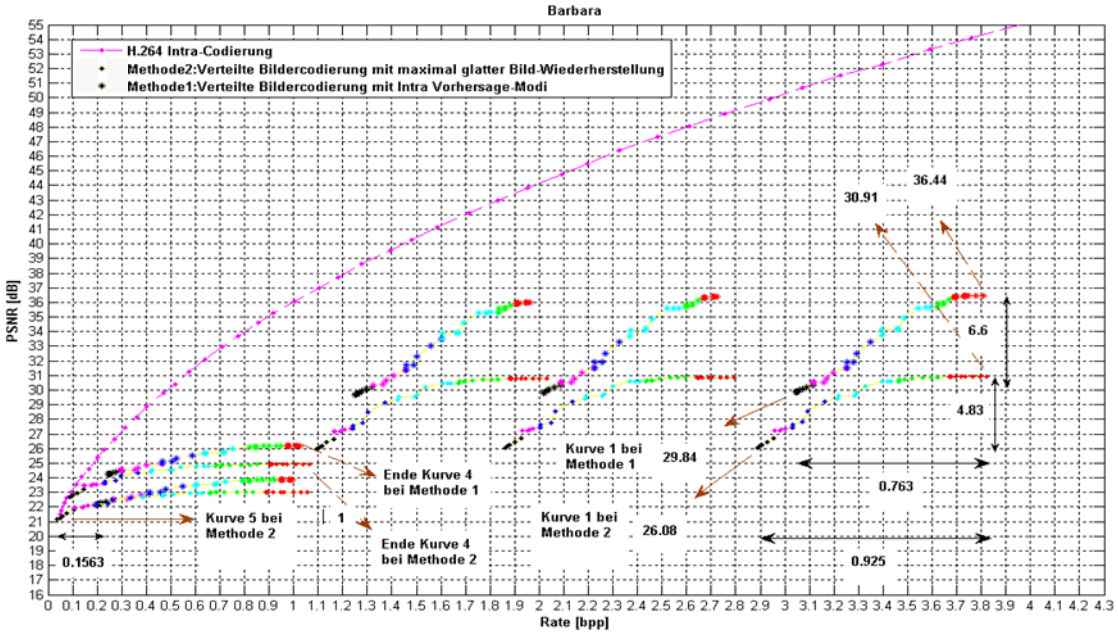


Abb. 6.28: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Barbara*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 sind qp=(0,10,20,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

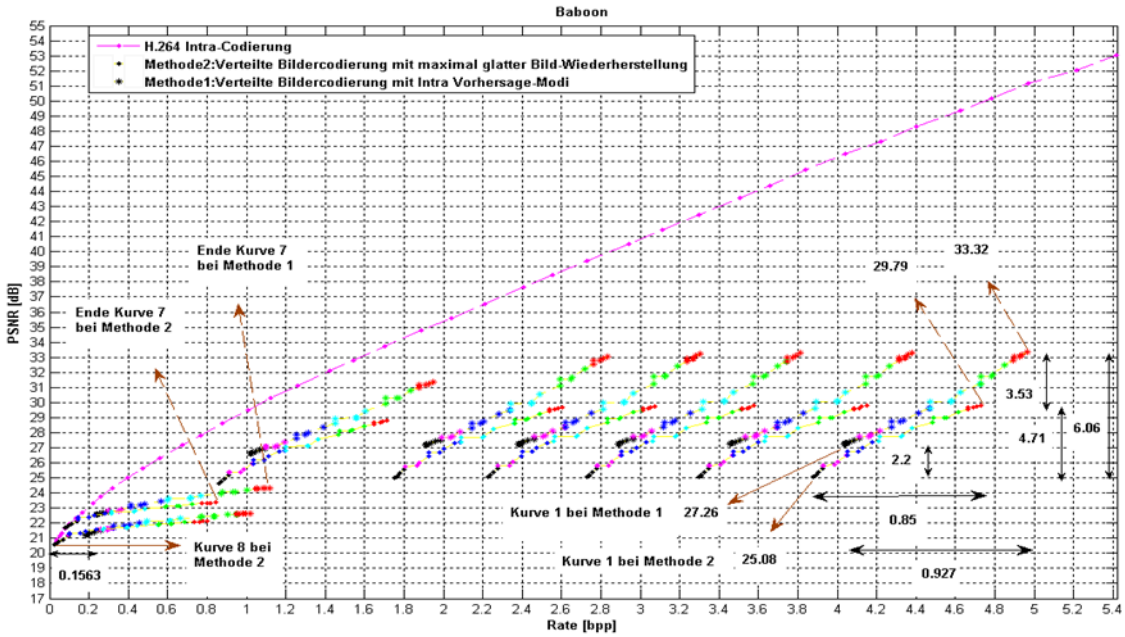


Abb. 6.29: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi für Bild *Baboon*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,30,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

Wir sehen auch, dass wenn die Qualität des Schlüssel-Bildes sehr niedrig ist (qp-Parameter über 35), die Verbesserung beim PSNR sehr klein ausfällt, und bei der schlechtesten Qualität des Schlüssel-Bildes (bei qp=50) nicht mehr als 2,56 dB. Der Grund dafür ist, dass die Vorhersage des WZ-Bildes am Decoder sehr ungenau ist und die LDPC-Decodierung nicht solch einen großen Vorhersagefehler (die Fehlerwahrscheinlichkeit ε ist groß) korrigieren kann. Unsere LDPC Codes können die Vorhersagefehler nur korrigieren, wenn ε kleiner/gleich 0.4917 ist (siehe Tab. 6.2). Man bemerke auch, dass im Bereich der niedrigen Qualität des Schlüssel-Bildes (wie bei qp größer als 35) ebenfalls keine große Verbesserung stattfindet und dass diese Verbesserung bei allen Bildern ähnlich gering ausfällt.

Wenn die Qualität des Schlüssel-Bildes sehr gut ist (für die meisten Bilder, wenn der qp-Parameter kleiner ist als 25), ist die Verbesserung beim PSNR hoch und liegt zwischen 4,71 dB und 8.11 dB. Die größte Verbesserung weist das Bild *Haus* auf. Die kleinste Verbesserung weist das Bild *Baboon* auf.

Wir sehen wie bei Methode I, dass für jedes Bild von einer bestimmten Qualität des Schlüssel-Bildes an keine Verbesserung (im Sinne des PSNR) bei einer weiteren Qualitätssteigerung mehr stattfindet. Diese spezifische Qualität liegt bei unseren Bildern etwa bei qp=15. Bei dieser Qualität zeigt der virtuelle Kanal, der sich zwischen den zu codierenden Bits und ihren vorhergesagten Bits befindet, auch bei weiteren Verbesserungen keine Verkleinerung der Fehlerwahrscheinlichkeit ε. Keiner unserer LDPC-Codes kann diesen großen Vorhersagefehler ε korrigieren (siehe Tab. 6.2). Wir haben aus der Tab. 6.2 ersehen, dass unsere LDPC-Codes die Vorhersagefehler nur dann korrigieren können, wenn der Wert des ε kleiner/gleich 0.4917 beträgt. Daher können wir diese Qualität bei verteilter Bildercodierung in der Methode II als Obergrenze bezeichnen.

Bei Nutzung einer Zusatzinformation von hoher Qualität (qp kleiner als 25) ist der Kurvenanstieg der ersten drei Pakete (DC Paket und AC1, AC2 Paket und AC3 bis AC5 Paket) bei der Methode II beim glatten Bild (wie Bild *Haus*) steil, bei einem weniger glatten Bild (wie Bild *Baboon*) dagegen gering. Beim Vergleich mit unserer ersten verteilten Codierungsmethode (Methode I in Abschnitt 6.2) sehen wir, dass der Kurvenanstieg der Methode II bei den ersten zwei Paketen (DC Paket und AC1, AC2 Paket) steiler als bei der Methode I ist; diese Unterschied wird aber kleiner, wenn das Bild weniger glatt ist (wie Bild *Baboon*). Der Grund dafür ist, dass die Methode I eine bessere Vorhersage für das WZ-Bild erzielt, wenn die Intra-Korrelation nicht niedrig ist.

Im Vergleich mit der Methode I (Abschnitt 6.2) erzielt Methode II bei Nutzung einer Zusatzinformation von niedriger Qualität (qp größer als 35) fast den gleichen PSNR nach der Korrektur der Transformationskoeffizienten. Wir sehen auch, dass Methode II vor der Korrektur der Transformationskoeffizienten durch Nutzung dieser niedrigen Qualität fast den gleichen PSNR wie H.264 Intra-Codierung erreichen kann.

Bei Nutzung guter Qualität der Zusatzinformation (qp kleiner als 15) erreicht die Methode I bessere PSNR als Methode II, der Grund dafür ist, dass die Transformationskoeffizienten des WZ-Bildes mit Methode I besser vorhergesagt werden.

Wir sehen in Abb. 6.30 und Abb. 6.31 das Maximum-optimum-Coding der Methode II im Vergleich zur Intra-Codierung H.264. Zu erkennen ist, dass bei allen Bildern der Anstieg der Optimum-coding-Kurve für hohe Bitrate klein ist (oft bei bpp größer als 0.6). Den Grund dafür haben wir oben erwähnt (die Obergrenze bei verteilter Bildercodierung in der Methode II).

Andererseits ist der Anstieg der Optimum-coding-Kurve bei niedrigen Bitraten groß: für Bild *Haus* und *Lena* im Bereich <0.4 bpp, für Bild *Barbara* für weniger als 0.5 bpp, für Bild *Baboon* im Bereich <0.6 bpp. Diesem großen Anstieg resultiert daraus, dass das vorhergesagte WZ-Bild ohne die vorherige Korrektur der Vorhersage-Fehler erreicht werden kann. Das heißt, wenn wir in diesem bpp-Bereich die Maximum-optimum-coding-Kurve erreichen wollen, sollten wir das Schlüssel-Bild in Intra-Coding H.264 besser quantisieren (durch Nutzung besserer Quantisierungsparameter QP). Wir sehen, dass bei vergrößertem bpp für das Bild *Haus* von 0.03 bpp bis 0.4 bpp, das PSNR auf fast 10.7 dB verbessern wird; für diesen gleichen Bereich von 0.03 bpp bis 0.4 bpp, wird das PSNR für das Bild *Lena* auf fast 6.58 dB verbessert, für das Bild *Barbara* auf fast 5.53 dB und für das Bild *Baboon* auf fast 3.64 dB. Das heißt, dass der Anstieg der Maximum-optimum-coding-Kurve bei diesem kleinen, erwähnten bpp-Bereich für das Bild *Haus* am höchsten wird, dagegen für das Bild *Baboon* am niedrigsten. Der Grund dafür ist, dass die Leistung der Intra-coding H.264 für ein sehr glattes Bild bei Nutzung feiner Quantisierung sehr hoch wird. Diese Leistung verschlechtert sich bei einem nicht glatten Bild (wie bei Bild *Baboon*).

Bei einer größeren Bitrate (für die meisten unsere Bilder ab 0.6 bpp) können wir die Maximum-optimum-coding-Kurve durch alleiniges Korrigieren der Vorhersagefehler erreichen, und zwar durch Nutzung der LDPC und des Feedback-Kanals. Im Vergleich zu Methode I sehen wir, dass die Methode II in einem niedrigen Bitrate-Bereich (für die meisten Bilder wenn bpp kleiner als 0.4) bessere PSNR liefert als Methode I. Das liegt an zwei Gründen.

1) Methode II korrigiert wenigere Bitebenen als Methode I und wie vorher erwähnt benötigt die Korrektur der MSBs im Vergleich zu LSBs wenigere Paritätsbits.
2) Der Encoder in Methode II verwendet keine *Suche des besten Vorhersage-Modus* (dieser Schritt war in Methode I, siehe Abb. 6.3). Das heißt, dass wir im Vergleich zur Methode I immer 0.1563 bpp weniger benötigen. Diese Einsparung war in diesem kleinen Bereich von bpp (bpp kleiner als 0.4) hilfreich.

Wir ersehen auch dass in Bereichen mit hohen bpp (für die meisten Bilder ab bpp=0.7) Methode I bessere PSNR als Methode II erreicht weil die Vorhersage Methode in Methode I besser als Methode II in diesem Bereich von bpp.

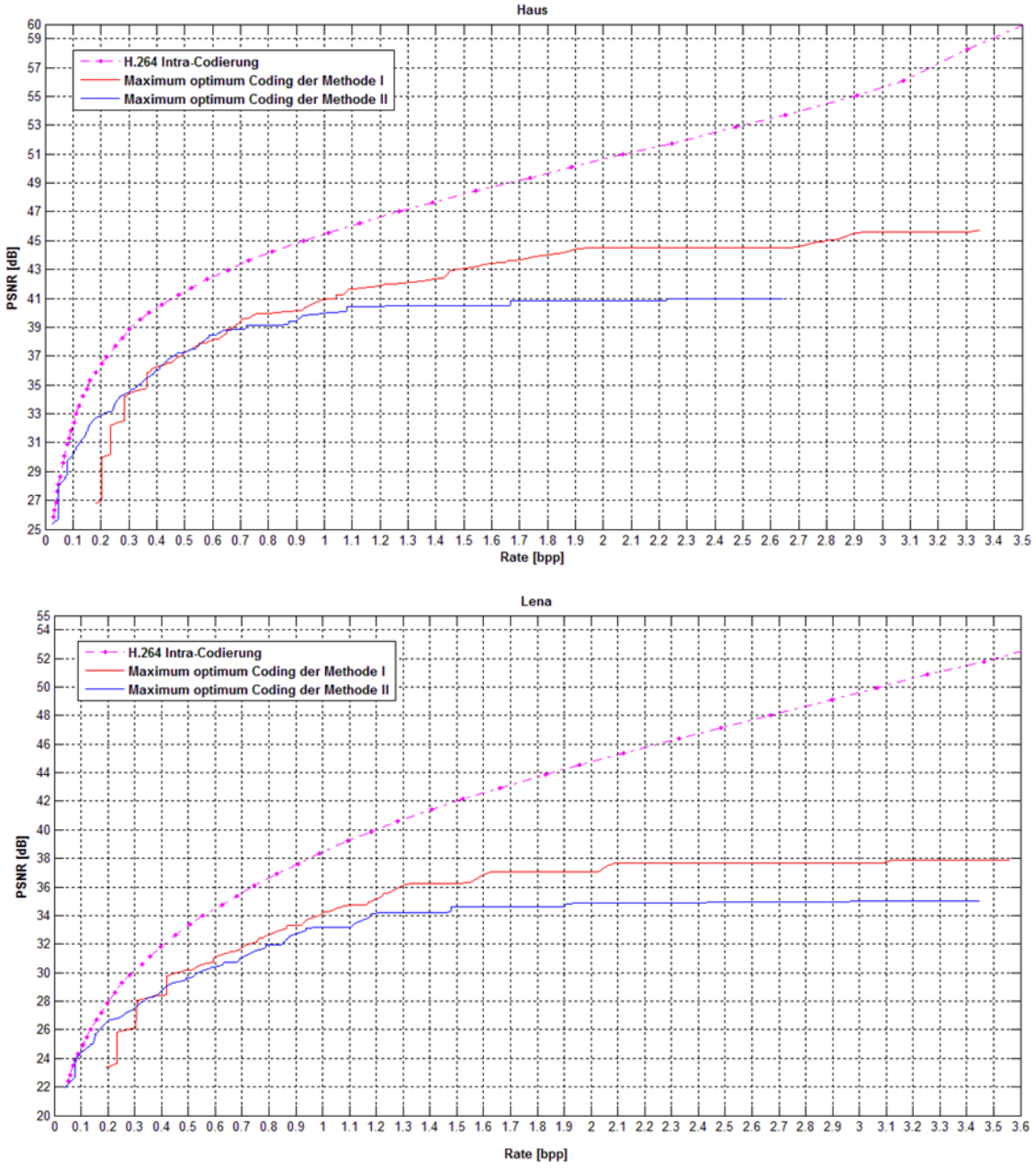


Abb. 6.30: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode II und der H.264 Intra-Codierung für die Bilder *Haus* (oben) und *Lena* (unten).

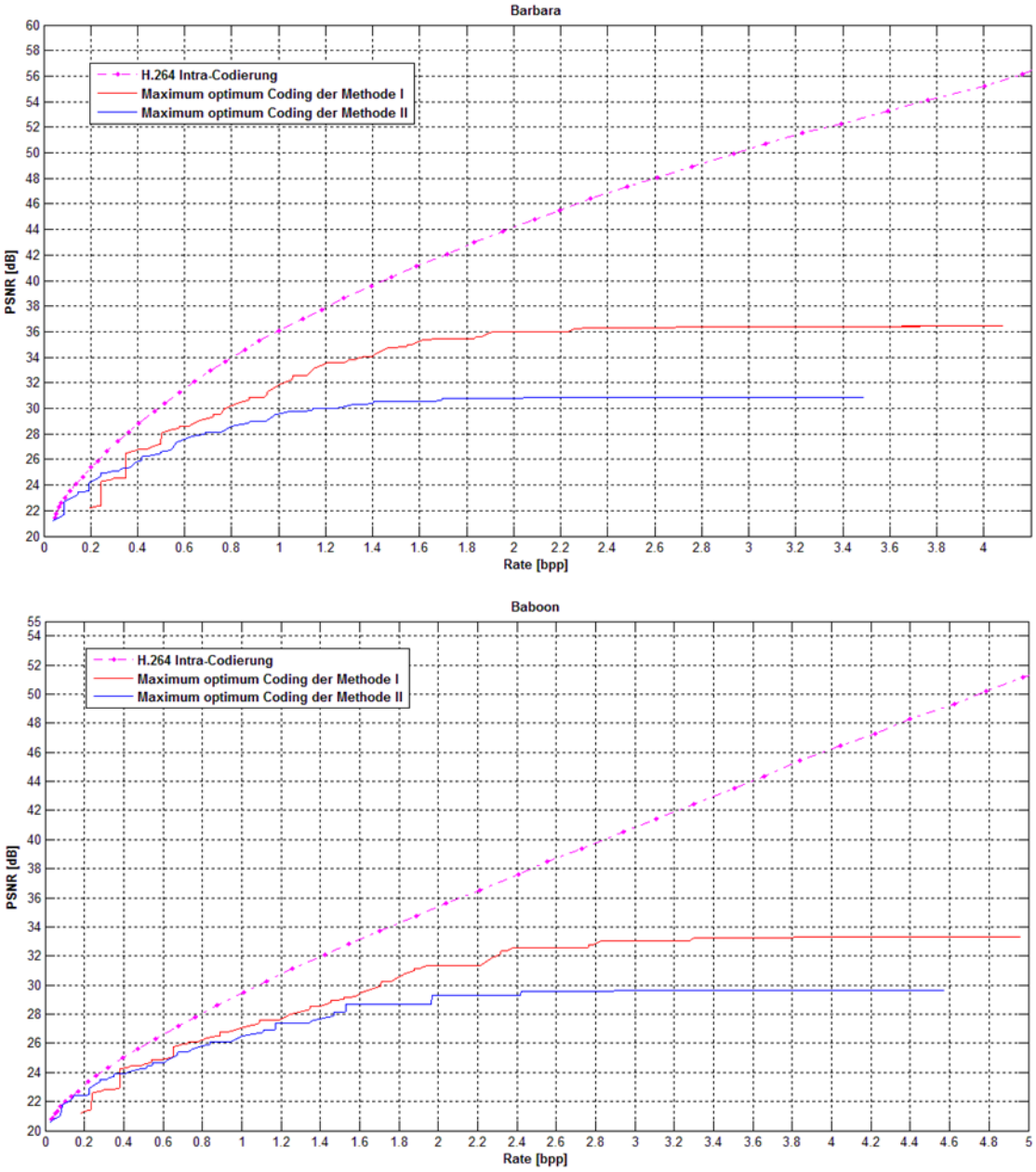


Abb. 6.31: Der Vergleich zwischen dem Maximum-optimum-Coding der Methode II und der H.264 Intra-Codierung für die Bilder *Barbara* (oben) und *Baboon* (unten).

6.3.3.2 Kompressionsfaktor und korrigierte Bit-Rate der Bitebenen

Wir sehen in Abb. 6.32 bis Abb. 6.35 den Kompressionsfaktor und die korrigierte Bit-Rate in der Methode II. Im Vergleich zu Methode I (siehe Tab. 6.9) sehen wir, dass für alle Bilder, mit Methode I mehr Bitebenen der niedrigen Transformationskoeffizienten als mit der Methode II korrigiert werden können (wir haben den Grund dafür vorher erklärt). Wir sehen auch, dass mit der Methode II bei den Bildern Barbara und Haus zufälligerweise mehr Bitebenen von manchen Hochfrequenz-Koeffizienten korrigiert werden können als mit der Methode I, aber von den PSNR-Kurven (siehe Abb. 6.26 und Abb. 6.28) sehen wir, dass bei der Korrektur dieser Bitebenen der PSNR sehr gering verberssert wird.

Wir sehen aus den Abbildungen wie bei Methode I, dass wir bei einer sehr guten Qualität der Zusatzinformation (für qp-Parameter 0 ist), vergleichsweise mehr Bitebenen korrigieren können als bei einer Zusatzinformation mit niedriger Qualität (qp 50 ist). Wir sehen auch, dass wenn ein Bild sehr glatt ist (wie das Bild *Haus*), wir bei Nutzung einer Zusatzinforation mit hoher Qualität (qp 0 ist) viel mehr Bitebenen der Koeffizienten korrigieren können (siehe Abb. 6.32).

Wir sehen wie bei Methode I, dass die ersten MSB_s der Hochfrequenz-Koeffizienten (letzte AC) üblicherweise sehr hohe Kompressionsfaktoren aufweisen. Wir ersehen auch, dass im Allgemeinen die signifikantesten Bitebenen (engl: most significant bit=MSB, manchmal auch MSB-1) eine höhere Kompression im Vergleich zu den am wenigsten signifikanten Bitebenen (engl: least significant bit-planes =LSB) haben.

Tab. 6.9: Vergleich der Anzahl der korrigierten Bitebenen der Koeffizienten zwischen Methode I und Methode II, wenn der Quantisierungsparameter des Schlüssel-Bildes qp=0 ist.

Koeffezient-Nummer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Haus-Meth I	8	8	8	6	5	6	6	6	6	6	5	6	5	6	6	5
Haus-Meth II	6	6	6	6	5	6	9	6	6	6	9	6	9	9	6	9
Lena-Meth I	6	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Lena-Meth II	5	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Barbara-Meth I	6	6	6	6	5	6	6	6	6	9	5	6	5	6	6	11
Barbara-Meth II	5	2	6	6	5	6	6	6	6	9	9	6	5	9	9	11
Baboon-Meth I	6	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Baboon-Meth II	4	5	5	4	5	6	6	6	6	6	5	6	5	6	6	5

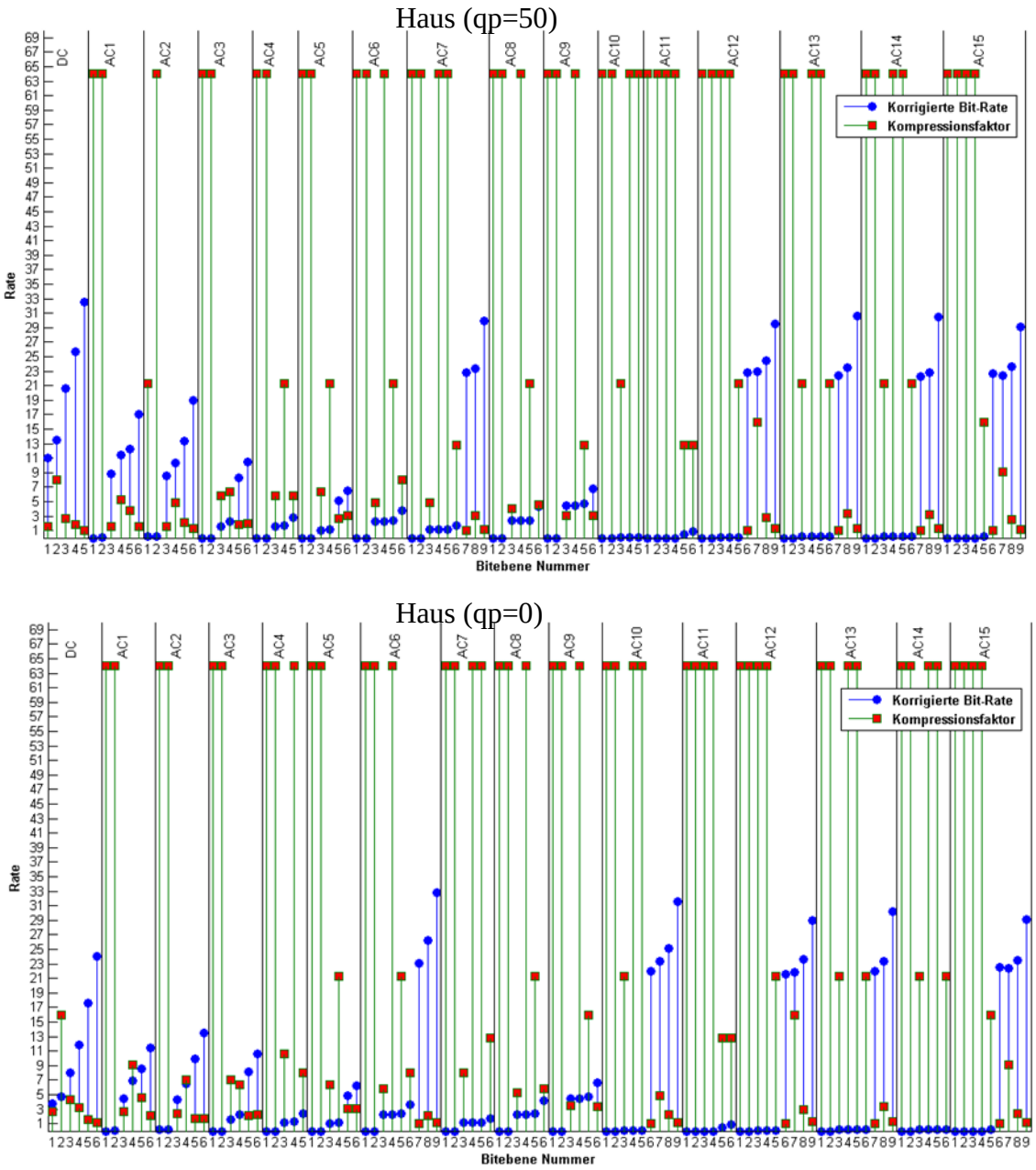


Abb. 6.32: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Haus* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

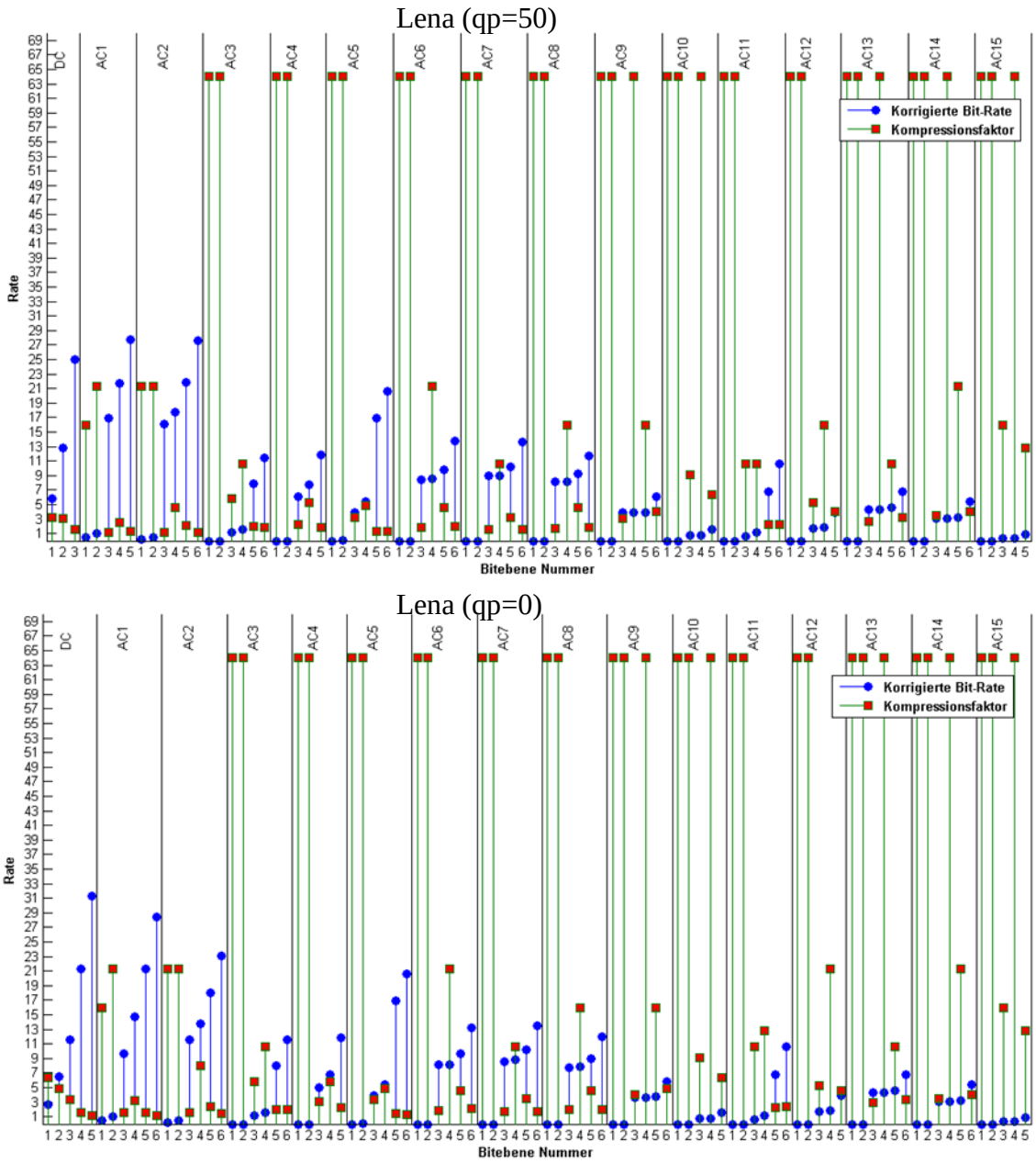


Abb. 6.33: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

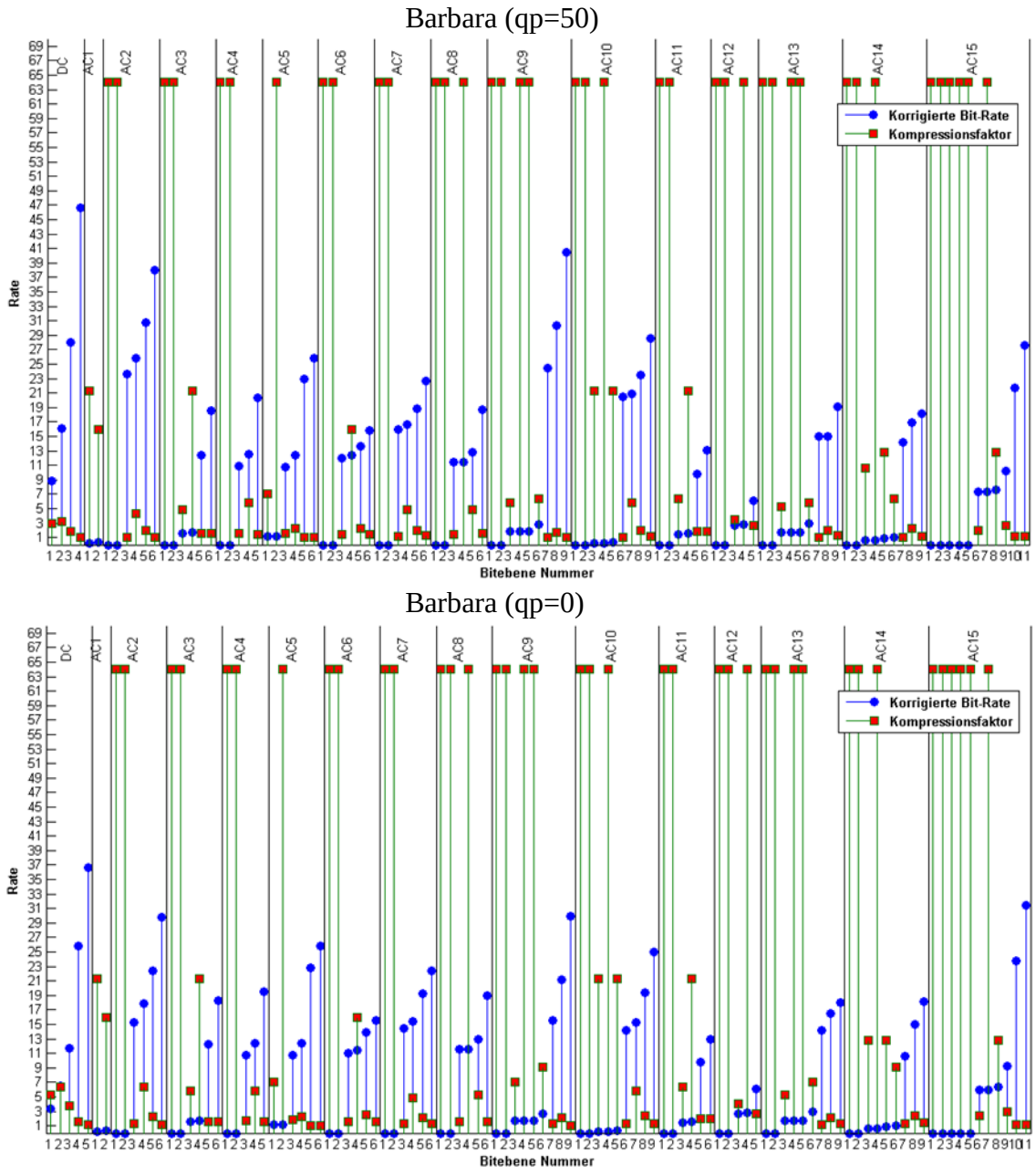


Abb. 6.34: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

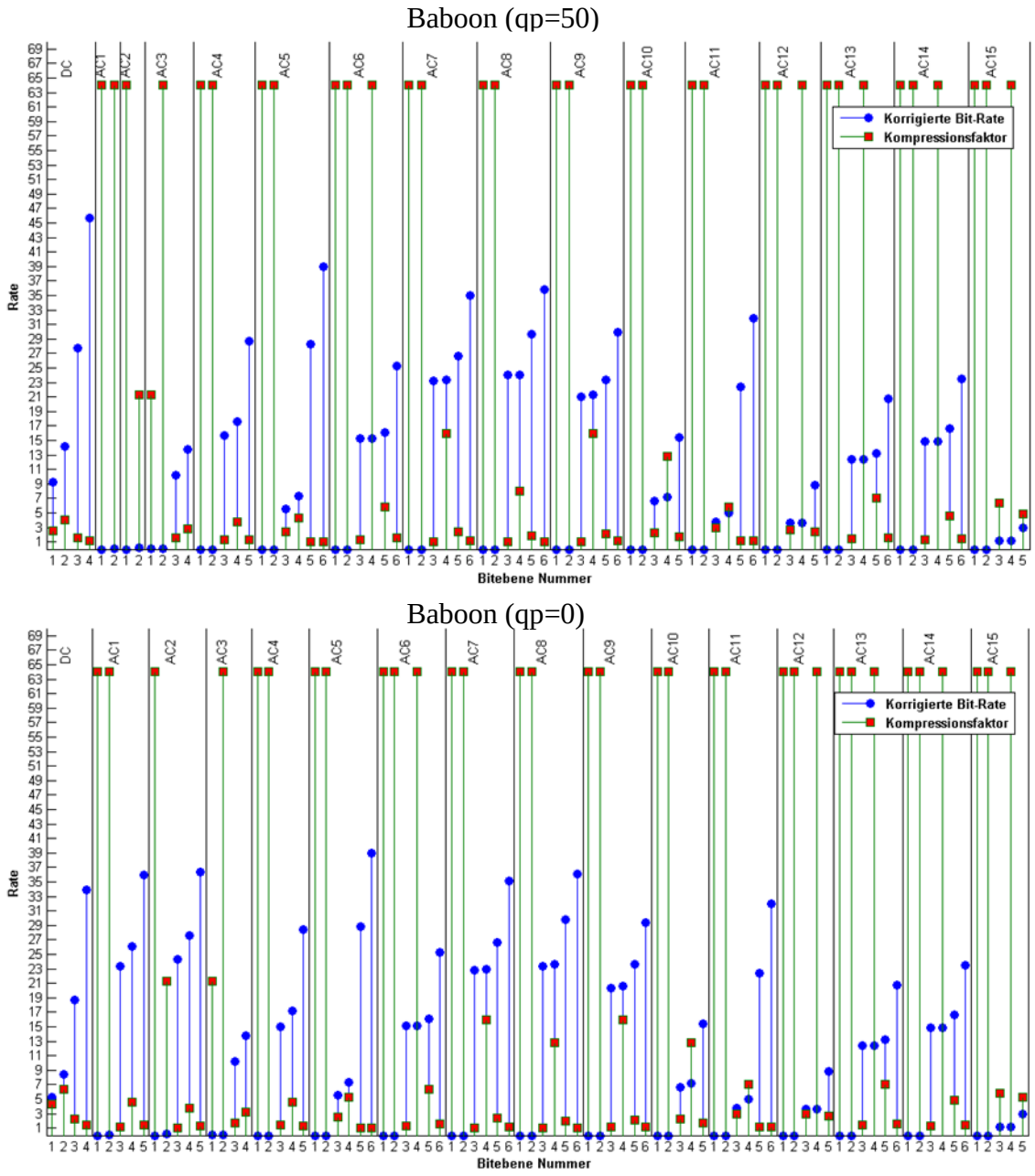


Abb. 6.35: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

6.3.3.3 Feedback-Kanal Rate

Der Feedback-Kanal hilft, wie worher erwähnt, dem Encoder dem kleinsten Paritätsvektoren zu finden, der für eine fehlerfreie LDPC-Decodierung ausreichend ist. Wir haben im Abschnitt 6.2.2 berechnet, dass die minimal Feedback-Kanal-Rate für die Korrektur einer Bitebene 1.5×10^{-5} bpp beträgt, und maximal 0.00096 bpp.

In der nächsten Tabelle sehen wir die Feedback-Kanal-Rate in bpp bei Anwendung der Methode II für das ganze Bild. Wir sehen, dass wenn der Quantizierungsparameter des Schlüssel-Bildes niedrig ist, die Anzahl der gesendeten Bits in dem Feedback-Kanal erhöht wird, weil unser Methode II mehr Bitebenen korrigiert, wenn die Zusatzinformation sehr gut ist (das ist der Fall, wenn die Qualität des Schlüssel-Bildes beim Decoder sehr gut ist). Außerdem sehen wir, dass das Bild *Haus* die geringste Feedback-Kanal-Rate zeigt; dagegen das Bild *Baboon* die höchste. Der Grund dafür ist, dass beim Bild *Haus* beim Decoder ein sehr gute vorhergesagte Bild vorhanden ist; beim Bild *Baboon* dagegen die schlechteste.

Im Vergleich zur Feedback-Kanal-Rate der Methode I (siehe Tab. 6.4) sehen wir, dass wir in Methode II eine ein bisschen geringere Rate haben, weil wir in Methode II wenigere Bitebenen korrigieren können als Methode I.

Tab. 6.10: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode II.

Qp (Quantisierungsparameter des Schlüssel-Bildes)	Haus	Lena	Barbara	Baboon
40	0.0007	0.0011	0.0018	0.0029
35	0.0010	0.0018	0.0033	0.0050
30	0.0012	0.0020	0.0036	0.0054
25	0.0016	0.0034	0.0050	0.0070
20	0.0020	0.0048	0.0065	0.0090
15	0.0033	0.0088	0.0106	0.0137

6.3.3.4 Wahrscheinlichkeitsdichtefunktion des Korrelationsrausches der Koeffizienten

Wir haben in Abschnitt 6.2.1 (siehe Codierungsratenvorhersage in Abschnitt 6.2.1) erwähnt, dass der Decoder das Korrelationsrauschen der Koeffizienten durch das Histogramm der Differenz zwischendem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes berechnet. Dann sendet der Decoder die Varianz σ^2 dieses Histogramms an den Encoder (*Codierungsraten-Vorhersage* in Abb. 6.25). Der Encoder nimmt an, dass die Verteilung zwischendem Koeffizienten-Band des originalen WZ-Bildes und dem Koeffizienten-Banddes vorhergesagten WZ-Bildes eine Laplacesche-Verteilungskurve (Laplacesche Wahrscheinlichkeitsdichtefunktion, engl:PDF) mit variance σ^2 ist. Wir sehen in Abb. 6.36 und Abb. 6.37 für Methode I und Methode II das Histogramm (für das Bild *Haus*) der Differenz zwischen den Koeffizienten-Bänder (DC, AC1, AC2) des originalen WZ-Bildes und den Koeffizienten-Bänder des vorhergesagten WZ-Bildes, außerdem zeigen wir die Laplacesche-Verteilungskurve für diese Koeffizienten-Bänder für Quantisierungsparameter qp=0 des Schlüssel-Bildes. Alle diese geannten Abbildungen sind auf denselben Bereich normiert.
Wir ersehen aus diesen Abbildungen, dass die Laplacesche Verteilung bei Verwendung der Methode II breiter ist (größere Vorhersagefehler) als bei der Verwendung der Methode I. Dies führt zur Korrekturvonwenigeren Bitebenen.

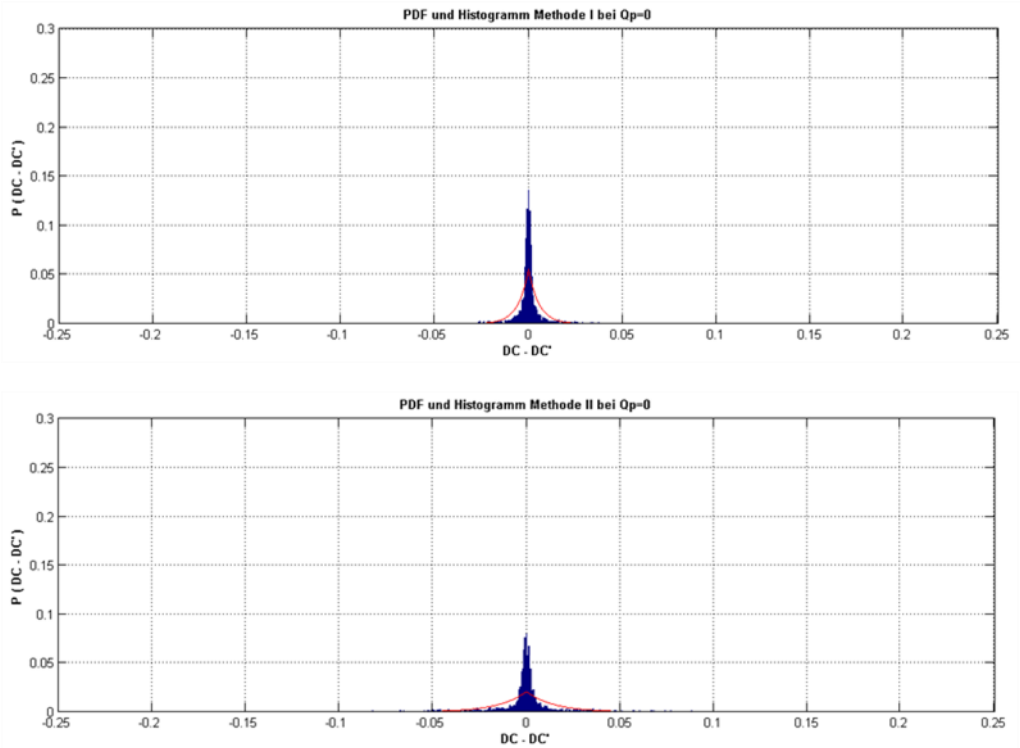


Abb. 6.36: Das Histogramm der Differez zwischen dem Koeffizienten-Band *DC* des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes sowie die PDF zwischen dem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode I und II berechnet.

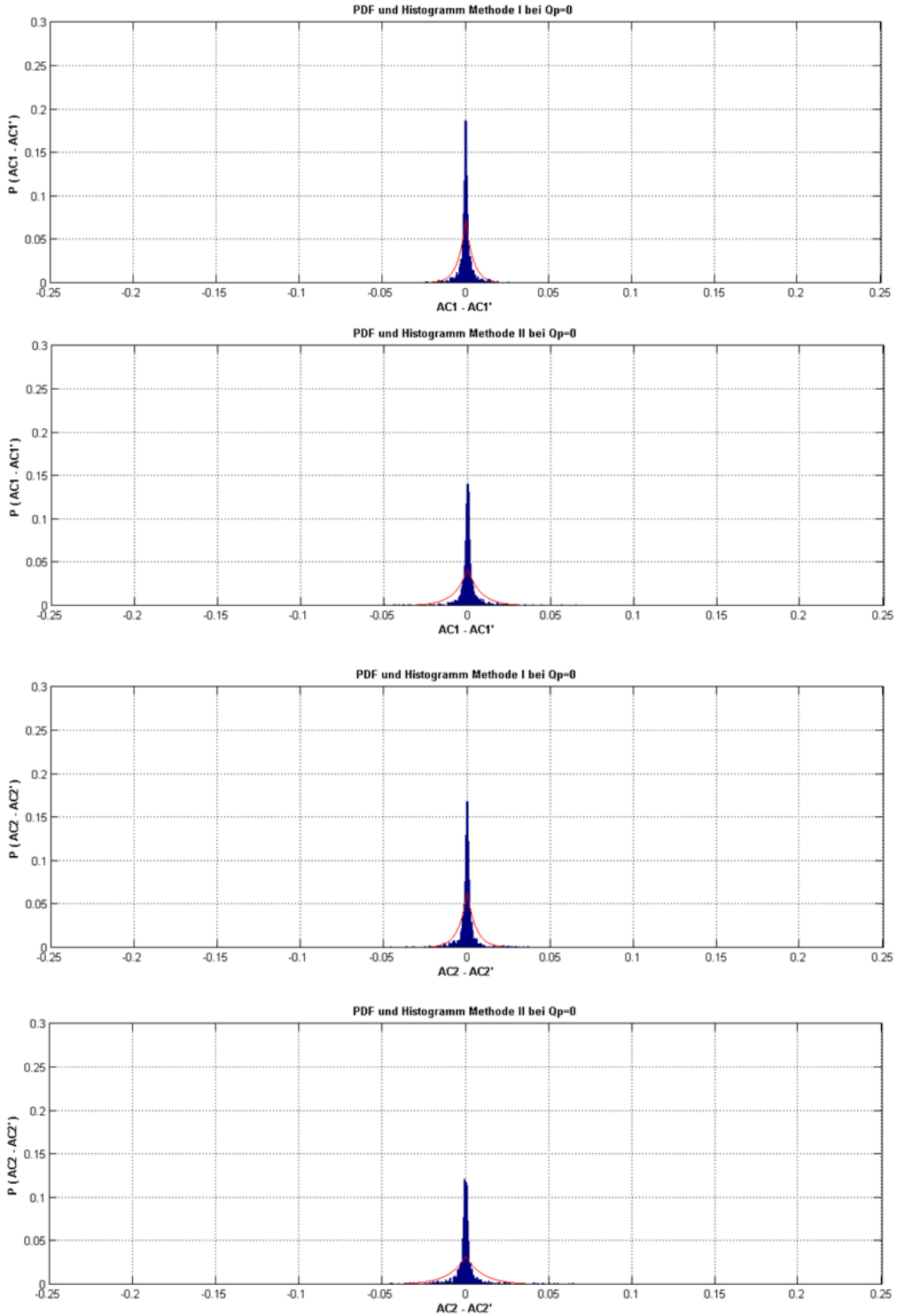


Abb. 6.37: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *AC1,AC2* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode I und II berechnet.

6.3.3.5 Komplexität der Encoder und der Decoder

Wir gehen von derselben Ausgangslage aus wie in Abschnitt 6.2.3.5 und benutzen dieselbe Art, um die Koplexität der Encoder und der Decoder zu berechenen.

Komplexität der Encoder:

In Tab. 6.11 und Tab. 6.12 sehen wir den Vergleich zwischen der Encoding-Zeit der Methode II und jene, wenn wir die Bilder mit H.264 Intra-Coding codieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter des Schlüssel-Bildes für Methode II, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Die Encoding-Zeit in der Methode II wie bei Methode I hängt besonders von dem Encoding des WZ-Bildes ab. Das heißt, wir brauchen im Vergleich zu H.264 Intra-Coding weniger Zeit mit der Methode II, weil wir die Hälfte des Bildes (das WZ-Bild) mit weniger Komplexität codieren. Das LDPC-Encoding ist viel weniger komplex als das H.264-Intra Coding. Verglichen mit der Encoding-Zeit der Methode I (siehe Tab. 6.5 und Tab. 6.6) sehen wir, dass die Methode II ein bisschen weniger Zeit braucht, weil der Encoder in Methode II die Vorhersage-Modi für die Blöcke des WZ-Bildes im Gegensatz zur Methode I nicht berechnet.

Tab. 6.11: Die Zeit der Codierung in Sekunden für die Bilder *Haus*, *Lena* in Methode II im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Coding in Sekunde		Methode II in Sekunde		Quotient:H.264 Intra-Coding/Methode II	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.875	1.016	0.527	0.608	1.66	1.67
35	0.968	1.203	0.579	0.716	1.67	1.68
30	1.094	1.5	0.655	0.882	1.67	1.7
25	1.297	1.907	0.776	1.115	1.67	1.71
20	1.594	2.375	0.954	1.397	1.67	1.7
15	2.078	3	1.229	1.744	1.69	1.72

Tab. 6.12: Die Zeit der Codierung in Sekunden für die Bilder *Barbara*, *Baboon* in Methode II im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Coding in Sekunde		Methode II in Sekunde		Quotient:H.264 Intra-Coding/Methode II	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	1.125	1.157	0.682	0.714	1.65	1.62
35	1.39	1.609	0.832	0.981	1.67	1.64
30	1.734	2.203	1.038	1.343	1.67	1.64
25	2.125	2.906	1.272	1.761	1.67	1.65
20	2.531	3.609	1.498	2.187	1.69	1.65
15	3.016	4.172	1.764	2.468	1.71	1.69

Komplexität der Decoder:

Wir werden auch hier wie bei der Codierungskomplexität die Decodierungskomplexität in Zeit messen.

In Tab. 6.13 und Tab. 6.14 sehen wir den Vergleich der Decodierung zwischen der Methode II und wenn wir die Bilder in H.264 Intra-Decoding decodieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter des Schlüssel-Bildes für Methode II, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Wie ersehen aus den Tabellen - wie zuvor bei der Betrachtung der Methode I -, dass die Decodierungskomplexität der Methode II viel höher ist als die des H.264 Intra-Decodings. Wir ersehen auch aus den Tabellen, dass wenn die Quantisierung des Schlüssel-Bildes sehr fein ist (wenn qp sehr niedrig ist), wird die Decodierungskomplexität höher.

Im Vergleich zur Methode I braucht die Methode II eine etwas geringere Decoding-Zeit, denn wie wir vorher erwähnt haben, können wir mit Methode II wenigere Biteben korrigeren.

Tab. 6.13: Zeit der Decodierung in Sekunde für die Bilder *Haus*, *Lena* in Methode II im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Decding in Sekunde		Methode II in Sekunde		Quotient:MethodeII/H.264 Intra-Decoding	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.047	0.062	19.76	37.93	420.4	611.8
35	0.062	0.094	34.95	85.82	563.8	913
30	0.078	0.125	44.73	114.21	573.5	913.7
25	0.110	0.187	81.98	235.47	745.3	1259.2
20	0.141	0.25	120.60	405.45	855.4	1621.8
15	0.219	0.359	241.01	873.61	1100.5	2433.5

Tab. 6.14: Zeit der Decodierung in Sekunde für die Bilder *Barbara*, *Baboon* in Methode II im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Decding in Sekunde		Methode II in Sekunde		Quotient:MethodeII/H.264 Intra-Decoding	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	0.079	0.093	79.8	156.2	1009.7	1679.8
35	0.109	0.156	171.2	385.5	1570.8	2471.4
30	0.156	0.25	274.3	663.8	1758.4	2655.2
25	0.219	0.344	493	1103.9	2251.4	3209.1
20	0.282	0.485	688	1668	2439.6	3439.2
15	0.391	0.625	1258.6	2513.2	3218.9	4021.1

6.3.3.6 Visueller Vergleich

Wir sehen in Abb. 6.38 das vorhergesagte Bild *Barbara* vor allen Korrekturen (bevor Paritätsbits vom Encoder gesendet wurden), codiert mit Methode II bei einer schlechten Qualität der Zusatzinformation (Quantisierungsparameter des Schlüssel-Bildes ist qp=40), und das gleiche Bild komplett codiert mittels H.264 Intra-Codierung bei fast gleicher Rate (bpp=0.202).

In Abb. 6.39 sehen wir links das gleiche Bild *Barbara* nach der Korrektur der Bitebenen bei gleicher Qualität der Zusatzinformation (qp=40), und wir sehen rechts das mit H.264 Intra-Codierung bei fast der gleichen Rate (bpp=1.101) komplett codierte Bild. So sehen wir in Abb. 6.39, dass das bei gleicher Rate zu codierende Bild bei Methode II im Vergleich zur Codierung des gesamten Bildes mit H.264 Intra-Codierung an Block-Artefakten und Verschwommenheit leidet.

Die Methode II leidet jedoch genau wie Methode I an Block-Artefakten, da wir das Bild nach dem Schachbrett-Muster aufgeteilt haben. Methode II besonders bei geringer bpp-Bereich (für die meisten Bilder wenn bpp kleiner als 0.4) führt aber zu stärker verschwommenen Bildern im Vergleich zu Methode I auf Grund der maximal glatten Bild-Wiederherstellung (siehe andere Beispiele in Anhangskapitel 8.4). Wir sehen im Anhangkapitel 8.2 auch, dass die Block-Artefakte offensichtlicher sind, wenn das Bild scharfe Abgrenzungen aufweist wie das Bild *Haus*.

Wir sehen auch im Anhangkapitel 8.2, dass kaum noch ein wahrnehmbarer Unterschied zwischen unserer verteilten Codierung bei sehr guter Qualität der Zusatzinformation und der H.264 Intra-Codierung besteht. Das heißt, dass die Methode II bei einer sehr hohen Bitrate für viele Anwendungung ausreichend ist.



Abb. 6.38: Das linke Bild ist das Vorhergesagte Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp = 40, PSNR dieses Bildes ist 24.18 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp.





Abb. 6.39: Das linke Bild ist das Vorhergesagte Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung Verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp =40, PSNR dieses Bildes ist 26.96 dB und die Rate 1.14 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp.

6.3.4 Zusammenfassung

Unser Ziel in diesr Methode II war es, eine verteilte Bild Codierung zu erstellen, welche das höchtmögliche PSNR zu erreichen versucht. Wir haben die eine Hälfte des Bildes (Schlüssel-Bild) in H.264 Intra-Coding und die zweite Hälfte (WZ-Bild) mit IntDCT Transform und LDPC Encoding codiert. In der IntDCT haben wir die beste Quantisierung benutzt. LDPC-Encoding codiert den Koeffizient-Band auf Bitebene, das heißt, wir codieren zuerst die MSBs, anschließend die LSBs. Der Decoder findet bei der Verwendung des decodierten Schlüssel-Bildes und bei der Verwendung maximal glatter *Bild-Wiederherstellung* das vorhergesagte WZ-Bild. Der Decoder nutzt LDPC-Decoding, um das vorhergesagte WZ-Bild zu korrigieren. Wir haben also eine Hybrid-Ratenkontrolle, um wenige Feedback-Kanal-Raten zu benötigen.

Wir haben gesehen, dass das Rate-Verzerrung-Verhältnis der Methode II unter dem Rate-Verzerrung-Verhältnis der Intra-Codierung des H.264 und auch unter der Rate-Verzerrung-Verhältnis der Methode I liegt, aber für eine niedrige Bit-Rate (bei den meisten Bildern wenn bpp kleiner als 0.2 ist) konnten wir die gleiche PSNR wie bei H.264 erreichen, und bei dieser niedrigen genannten Bit-Rate konnten wir nicht so viele Bitebene korrigieren (keine Verbesserung in PSNR durch Senden eines längeren Paritätsvektoren): Wir haben auch herausgefunden, dass wir bei sehr hoher Bit-Rate die PSNR stark verbessern konnten, aber die Verbesserung ist geringer als die Verbessurung in Methode I. Der wahrnehmbare Unterschied zwischen der Methode II und dem H.264 Intra-Coding bei sehr hoher Bit-Rate ist nicht groß. Deswegen lohnt es sich, für manche Anwendungen die Methode II statt H.264 Intra-Coding zu benutzen.

Es ist bei der Durchführung der Methode II besonders wichtig, dass die Codierungskomplexität geringer ist als die Komplexität des H.264 Intra-Coding und auch etwas geringer als bei der Anwendung der Methode I. Die Decodierungskomplexität ist wie bei Methode I sehr viel höher als die des H.264 Intra-Decoding. Wir haben auch gesehen, dass die Feedback-Kanal-Rate wie bei Methode I sehr gering ist und etwas niedriger als Methode I liegt.

6.4 Methode III:Verteilte Bildercodierung mit linearer Pixel-Vorhersage

In diesem Abschnitt wird eine Wyner-Ziv Bildercodierung namens verteilte Bildercodierung mit Nutzung linearer Pixel-Vorhersage vorgestellt, angewendet und im Detail bewertet. Die Ergebnisse in diesem Abschnitt zeigen, dass das Rate-Verzerrung-Verhältnis dieser Methode ebenfalls unter dem Rate-Verzerrung-Verhältnis der Intra-Codierung H.264 liegt. Wir werden sehen im vergleich zu Methode I und II, dass Methode III das bestens Rate-Verzerrung-Verhältnis aufweist, wenn das zu codierende Bild sehr glatt ist. Die visuelle Wahrnehmung in Methode III ist besonders bei Nutzung niedriger bpp schlechter als in Methode I und Methode II. Die Codierungskomplexität bleibt geringer als beim H.264 Intra-Coding - wie bei der Methode II, aber die Decodierungskomplexität bleibt höher als die Komplexität der H.264 Decoder - wie bei den Methoden I und II.

Wir haben gesehen, dass das Bild in Methode I und Methode II nach dem Schachbrett-Muster aufgeteilt wurde. Wir nehmen an, dass ein fehlender Pixel in einem sehr glatten Bild besser vorherzusagen ist als ein fehlender Block in einem vorherzusagenden Bild. Deswegen werden wir versuchen, in dieser Methode kein Schachbrett-Muster, sondern ein Spalten-Muster zu benutzen; der Decoder nutzt neben der linearen Pixel-Vorhersage dieses Muster, um bessere PSNR zu erreichen als mit Methode I und Methode II.

6.4.1 Codierung und Decodierung

In Abb. 6.40 ist der Aufbau dieser Methode dargestellt. Hier wird, wie bei Methode I und Methode II, das Bild in zwei Teile (WZ-Bild und Schlüssel-Bild) aufgeteilt, aber die Aufteilung erfolgt wie in Abb. 6.41 vorgegeben. Die Größe von Bild I (WZ-Bild) entspricht der halben Größe des ganzen Bildes. Wenn das Originalbild die Größe $N \times N$ hat, sind das WZ-Bild und das Schlüssel-Bild jeweils $N \times \frac{N}{2}$ groß.

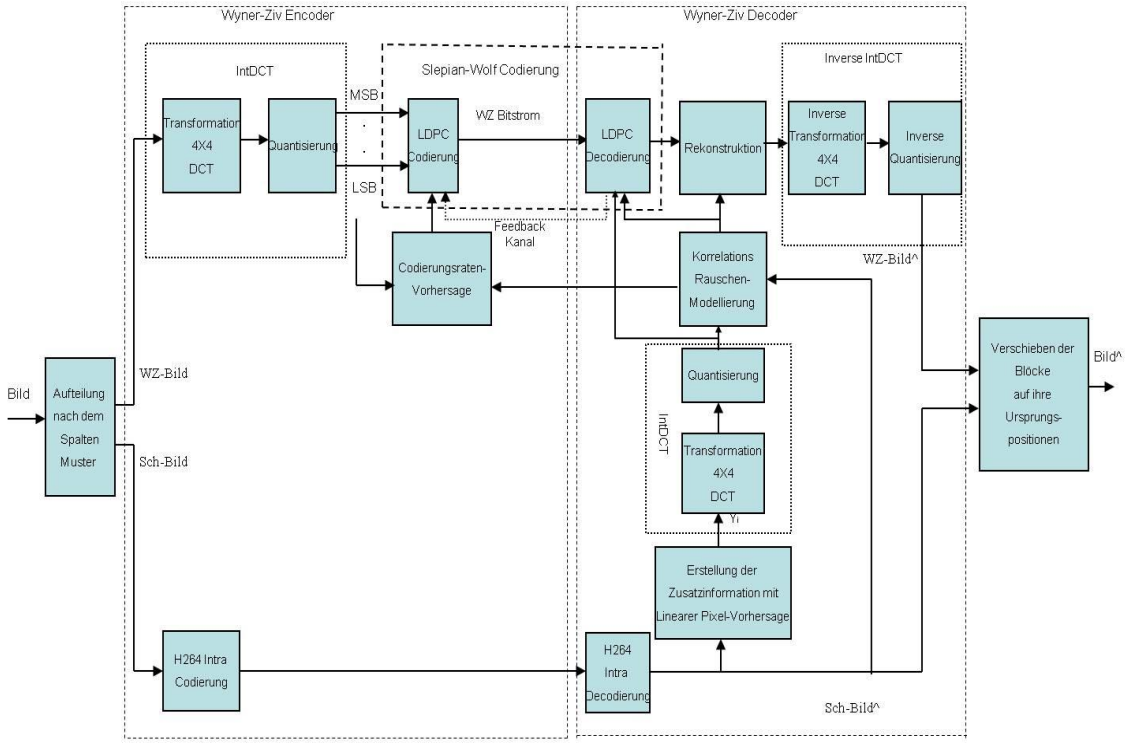


Abb. 6.40: Blockdiagramm der verteilten Bildercodierung mit linearer Pixel-Vorhersage.

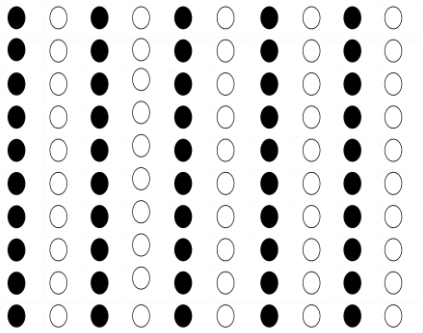


Abb. 6.41: Das Spalten-Muster nach dem das Bild in WZ-Bild (weiße Spalten) und Schlüssel-Bild (schwarze Spalten) aufgeteilt wird.

Wir codieren das WZ-Bild mit einem Wyner-Ziv-Encoder und mit LDPC, das Schlüssel-Bild hingegen mittels herkömmlicher H.264 Intra-Codierung. Der Encoder sendet an den Decoder keine Information, die diesem hilft, die beste Vorhersage für das WZ-Bild zu finden.

Jeder Block beim Encoder in Abb. 6.40 ist identisch mit dem gleichen Block in Abb. 6.3, außer dem *Aufteilungs-Block*. Der Decoder nutzt nur die lineare Pixel-Vorhersage, um das WZ-Bild vorherzusagen.

Lineare Pixel-Vorhersage:

Wir benutzen eine eindimensionale Daten-Interpolation, um die Pixel-Werte des WZ-Bildes vorherzusagen. Diese Vorhersage ist nichts anders als der Miitelwert der Pixel links und rechts vom dem fehlenden, noch zu berechnenden Pixel. Es gibt weitere Daten-Interpolation, aber wir entscheiden uns für diese Vorhersage-Methode, weil sie zu unserem Spalten-Muster passt und auch weil sie nicht komplexer ist als im Vergleich zu anderen Vorhersage-Methoden.

6.4.2 Bewertung der Ergebnisse

6.4.2.1 PSNR

In Abb. 6.46 stellen wir das Originalbild (*Barbara*) sowie das WZ-Bild und das Schlüssel-Bild dar. Die anderen (*Haus*, *Lena*, *Baboon*) sind im Anhangkapitel 8.3.1 zu sehen. In den Abbildungen Abb. 6.42 bis Abb. 6.45 sehen wir den PSNR-Vergleich zwischen Mehthode I, Mehthode II, Mehthode III und der herkömmlichen H.264 Intra-Codierung. Es ist ersichtlich, dass die dritte verteilte Methode (Method III) das beste Rate-Verzerrungs-Verhältnis dann aufweist, wenn das Bild sehr glatt ist (wie Bild *Haus*).

Wir sehen wie bei Methode I und Methode II, dass das vorhergesagte Bild *Haus* im Vergleich mit den anderen Bildern das größte PSNR erreicht. Dieser Vergleich stimmt bei Anwendung des gleichen Quantisierungsparameters qp des Schlüssel-Bildes. Wir haben vorher erwähnt, dass das vorhergesagte Bild dem ersten schwarzen Punkt jeder PSNR-Kurve entspricht. Wir sehen auch, dass wie mit Methode I und II die Verbesserung beim PSNR sehr klein ausfällt, wenn die Qualität des Schlüssel-Bildes sehr niedrig ist (qp-Parameter von H.264 über 35), dass die Verbesserung des PSNR bei der schlechtesten Qualität des Schlüssel-Bildes (bei qp=50) nicht mehr als 2 dB beträgt. Wenn die Qualität des Schlüssel-Bildes sehr gut ist (für die meisten Bilder, wenn der qp-Parameter von H.264 kleiner ist als 20), ist die Verbesserung beim PSNR hoch und liegt zwischen 3.41 dB und 6.35 dB. Wir sehen wie bei Methode I und II, dass für jedes Bild von einer bestimmten Qualität des Schlüssel-Bildes an bei einer weiteren Qualitätssteigerung keine Verbesserung (im Sinne des PSNR)mehr stattfindet. Diese spezifische Qualität liegt bei unseren Bildern etwa bei qp=20. Wir bezeichnen diese Qualität als Obergrenze bei verteilter Bildercodierung in Methode III. Im Vergleich mit Methode I, Methode III das besste Rate-Verzerrung-Verhältnis wenn das Bild sehr glatt ist (wie das Bild *Haus*). Das ist weil in Methode III für solchen Bilder besseres vorhergesagtes WZ-Bildes. Im Vergleich mit Methode II erzielt Methode III bei Nutzung sehr hoch Qualität der Zusatzinformation (qp-Parameter von H.264 über 35) oft bessere PSNR (außer Bild *Barbara*). Das ist weil in Methode III das Vorhergesagtes WZ-Bildes oft besser als mit Methode II. Bei Methode III können wir für das Bild Barbara viele Bitebenen der Koeffizienten korrigieren, aber diese korrigierten Bitebenen verbessern den PSNR jedoch nicht wesentlich und die Kosten (in bpp) für diese Verbesserung sind hoch.

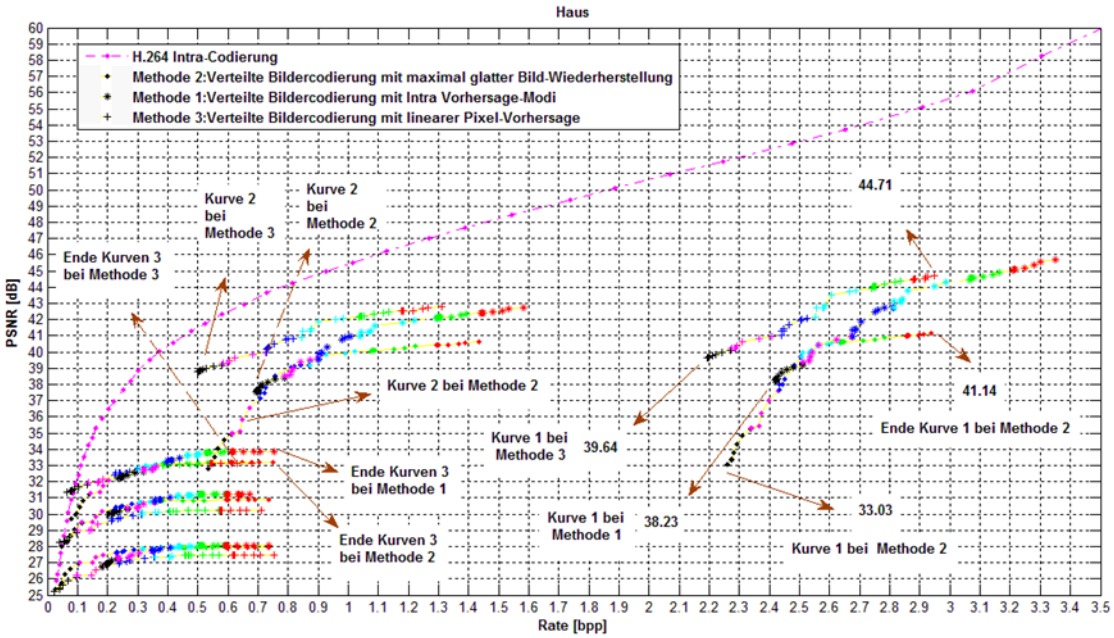


Abb. 6.42: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Haus*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 qp=(0,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

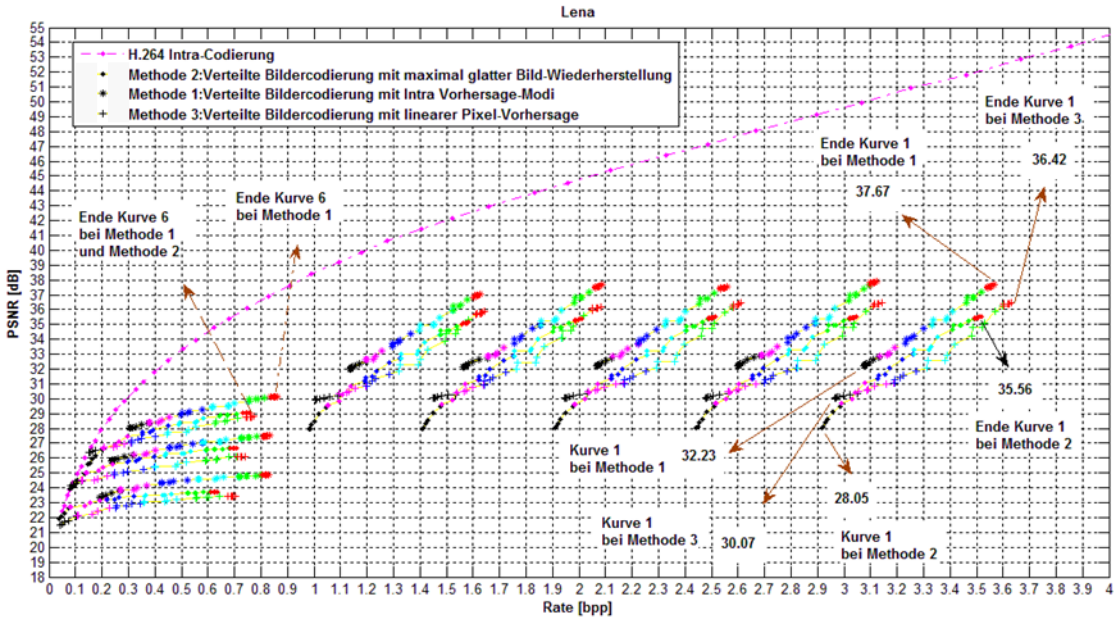


Abb. 6.43: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Lena*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,40,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

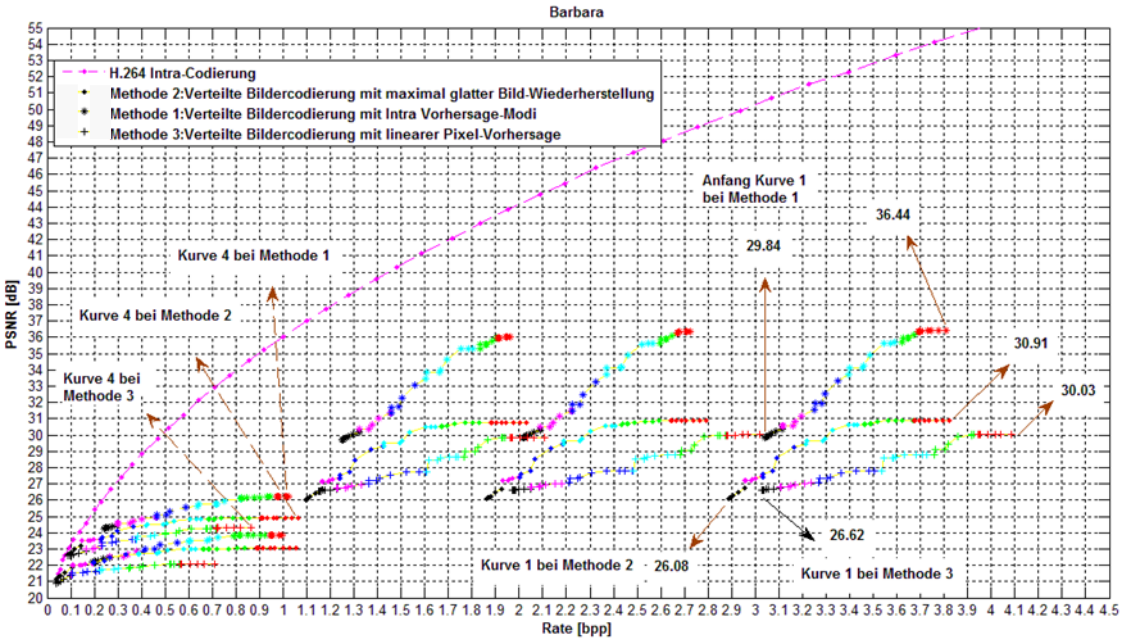


Abb. 6.44: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Barbara*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 5 sind qp=(0,10,20,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.

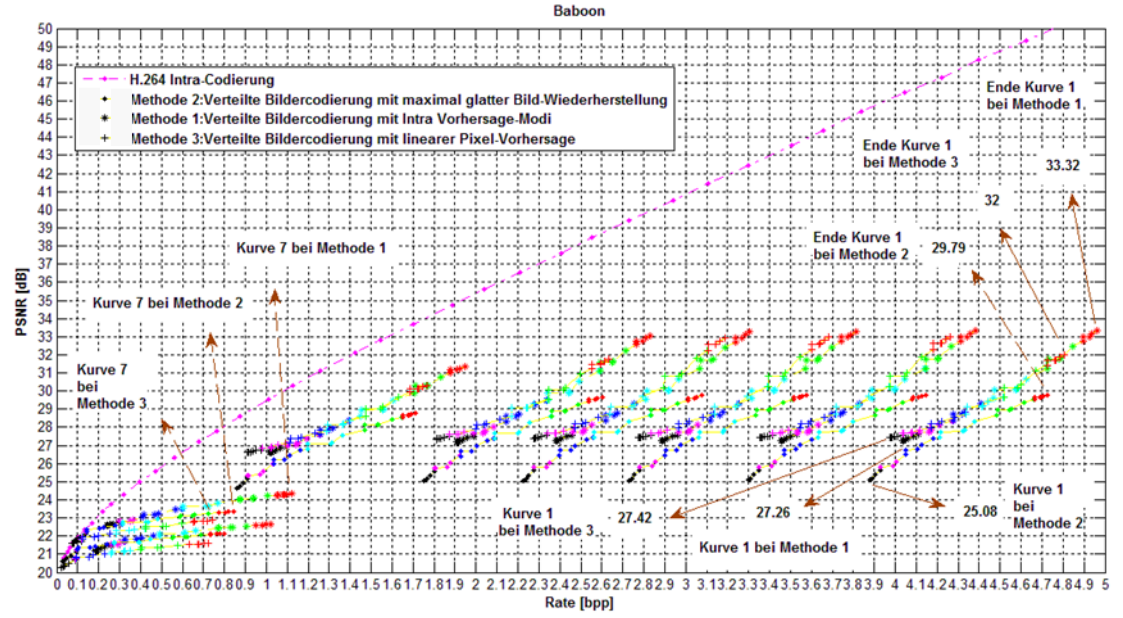


Abb. 6.45: Vergleich zwischen H.264 Intra-Codierung und verteilter Bildercodierung mit linearer Pixel-Vorhersage, maximal glatter Bild-Wiederherstellung, mit Intra-Vorhersage-Modi, für Bild *Boaboon*. Der Quantisierungsparameter des Schlüssel-Bildes von Kurve 1 bis Kurve 8 sind qp=(0,5,10,15,20,30,45,50). Jede Kurve beginnt mit schwarzen Markern. Die schwarzen Marker stehen für die Korrekturen der DC-Koeffizienten dar, Magenta für AC1 und AC2, Blau für AC3 bis AC5, Cyan für AC6 bis AC9, Grün für AC10 bis AC13, Rot für AC14 und AC15.



Abb. 6.46: Originalbild (*Barbara*), WZ-Bild und Schlüssel-Bild bei Spalten-Aufteilung.

Wir sehen in Abb. 6.47 und Abb. 6.48 das Maximum-optimum-Coding der Methode III im Vergleich zur Intra-Codierung H.264. Zu erkennen ist, dass bei allen Bildern der Anstieg der Optimum-coding-Kurve bei hohe Bitrate klein ist (oft bei bpp größer als 0.6). Den Grund dafür haben wir oben erwähnt (die Obergrenze bei verteilter Bildercodierung in unserer Methode III). Auf der anderen Seite ist der Anstieg der Optimum-coding-Kurve bei niedrigen Bitraten (für die meisten Bilder im Bereich <0.4 bpp) wie bei Methode I und II groß. Diesem großen Anstieg resultiert daraus, dass das vorhergesagte WZ-Bild ohne die vorherige Korrektur der Vorhersage-Fehler erreicht werden kann. Das heißt, wenn wir in diesem bpp-Bereich die Maximum-optimum-coding-Kurve erreichen wollen, sollten wir das Schlüssel-Bild in Intra-Coding H.264 besser quantisieren (durch Nutzung besserer Quantisierungsparameter QP). Bei einer größeren Bitrate (für die meisten unsere Bilder ab 0.6 bpp) können wir die Maximum-optimum-coding-Kurve durch alleiniges Korrigieren der Vorhersagefehler erreichen, und zwar durch Nutzung der LDPC und des Feedback-Kanals.

Im Vergleich zu Methode I und II sehen wir, dass wir nur bei einem sehr glatten Bild die beste Rate-Verzerrung-Kurve erreichen, so wie bei Bild *Haus*. Der Grund dafür ist, dass die Lineare Pixel-Vorhersage besser ist als die anderen Vorhersage-Methoden in Methode I und II für Bilder solcher Art. Deshalb können wir oft mehr Bitebenen korrigieren als mit Methode I und Methode II.

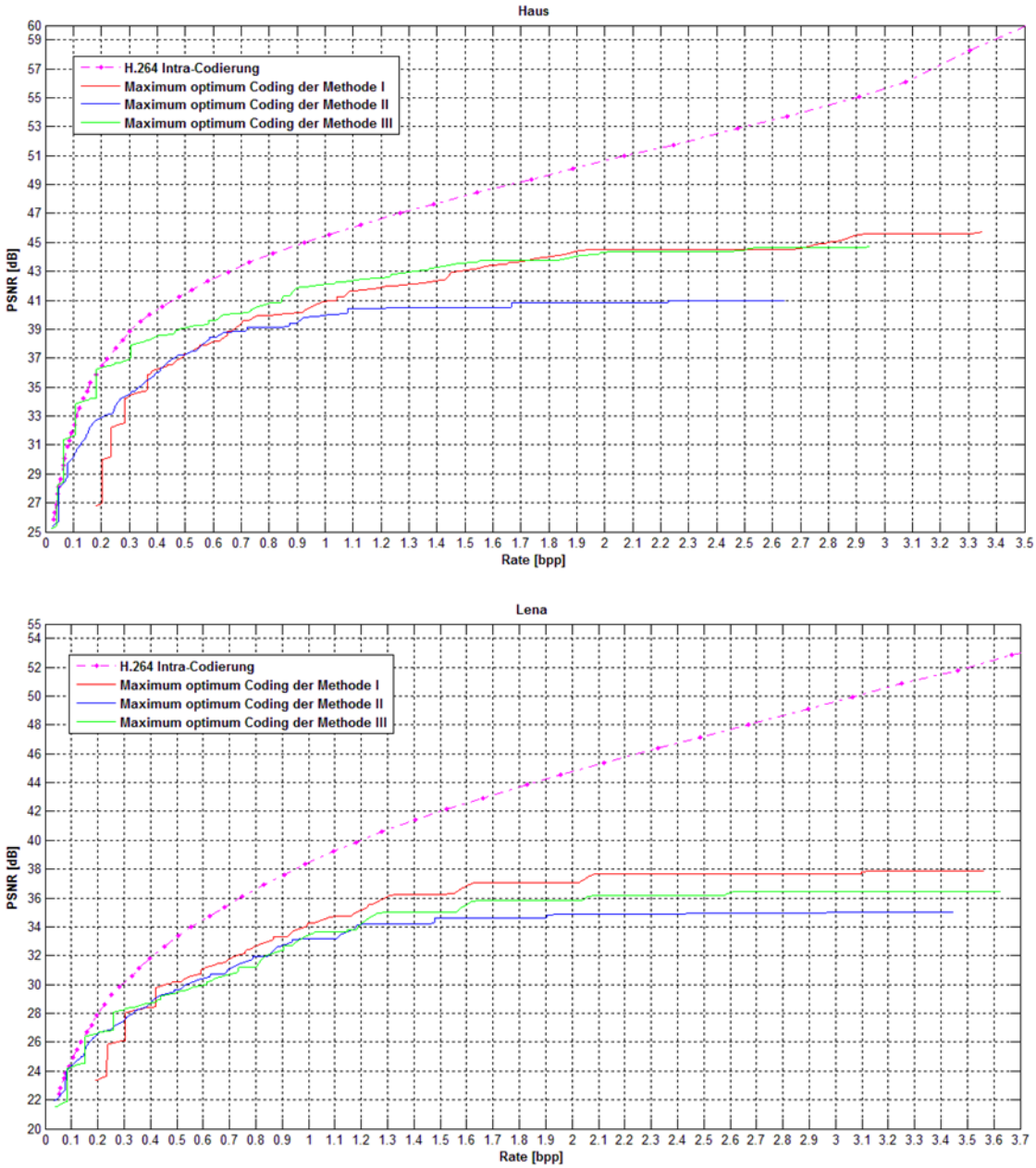


Abb. 6.47: Vergleich zwischen Maximum optimum Coding der Methode III und H.264 Intra-Codierung und Methode I und Mthode II für die Bilder *Haus* (oben) und *Lena* (unten).

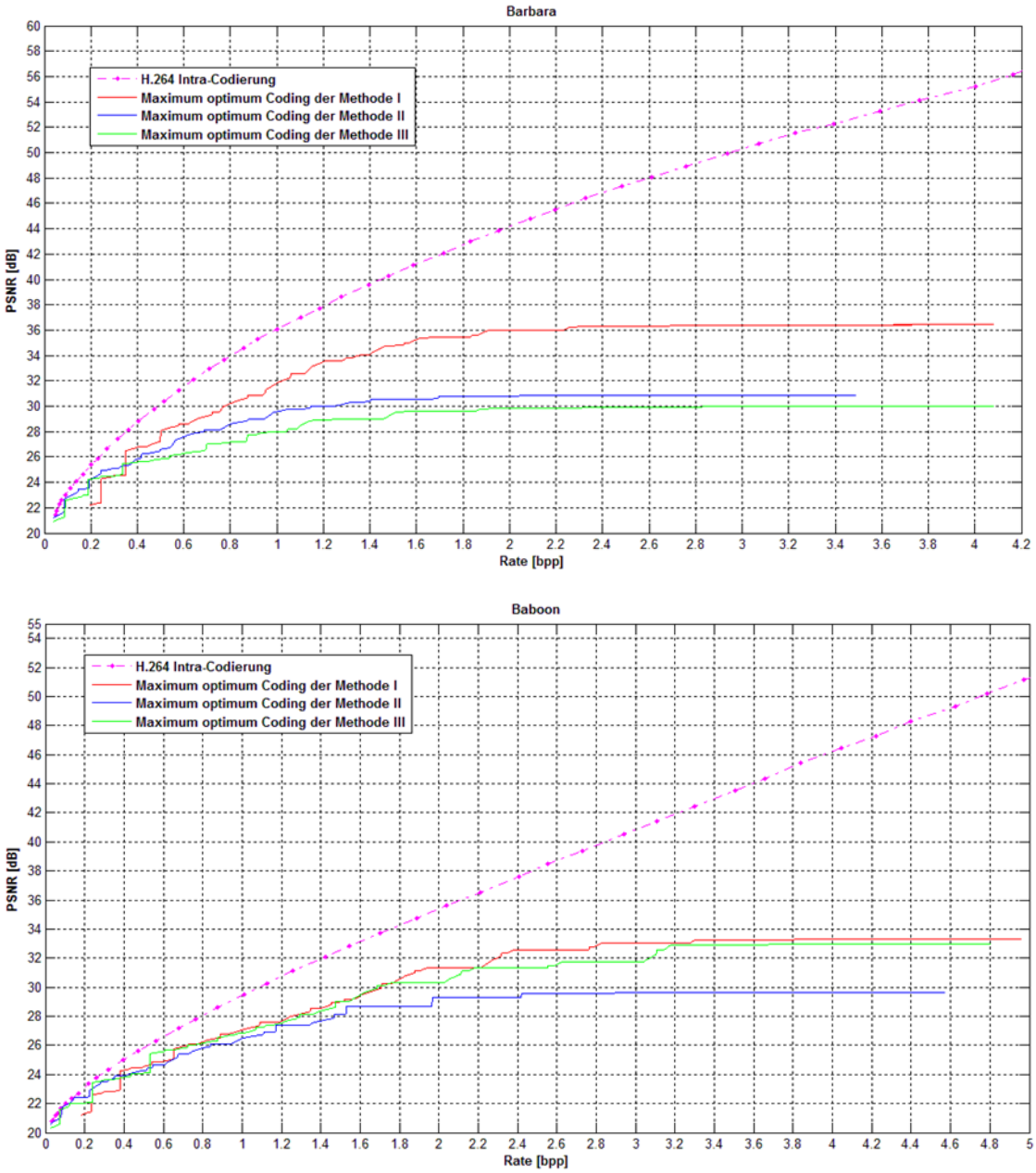


Abb. 6.48: Vergleich zwischen Maximum optimum Coding der Methode III und H.264 Intra-Codierung und Methode I und Mthode II für die Bilder *Barbara* (oben) und *Baboon* (unten).

6.4.2.2 Kompressionsfaktor und korrigierte Bit-Rate der Bitebenen

Im Vergleich zur ersten und zweiten Methode (siehe Tab. 6.15) sehen wir, dass für alle Bilder, mit Methode III mehr Bitebenen der niedrigen Transformationskoeffizienten korrigiert werden können (wir haben den Grund dafür vorher erklärt). Wir sehen in Abb.6.49 bis Abb. 6.52 den Kompressionsfaktor und die korrigierte Bit-Rate in der Methode III. Wir sehen wie bei Methode I und II, dass wir bei einer sehr guten Qualität der Zusatzinformation (der Quantisierungsparameter des Schlüssel-Bildes qp 0 ist), vergleichsweise mehr Bitebenen korrigieren können als bei einer Zusatzinformation mit niedriger Qualität (qp 50 ist). Wir sehen auch in den Abbildungen, dass wenn ein Bild sehr glatt ist (wie das Bild *Haus*), wir bei Nutzung einer Zusatzinforation mit hoher Qualität (qp 0 ist) viel mehr Bitebenen der Koeffizienten korrigieren können.

Wir sehen wie bei Methode I und II, dass die ersten MSB_s der Hochfrequenz-Koeffizienten (letzte AC) üblicherweise sehr hohe Kompressionsfaktoren aufweisen. Wir ersehen auch, dass im Allgemeinen die signifikantesten Bitebenen (engl: most significant bit=MSB, manchmal auch MSB-1) eine höhere Kompression im Vergleich zu den am wenigsten signifikanten Bitebenen (engl: least significant bit-planes =LSB) haben.

Tab. 6.15: Vergleich der Anzahl der korrigierten Bitebenen der Koeffizienten zwischen Methode I, Methode II und Methode III, wenn der Quantisierungsparameter des Schlüssel-Bildes qp=0 ist.

Koeffezient-Nummer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Haus-Meth I	8	8	8	6	5	6	6	6	6	6	5	6	5	6	6	5
Haus-Meth II	6	6	6	6	5	6	9	6	6	6	9	6	9	9	6	9
Haus-Meth III	8	8	9	6	9	6	6	9	9	6	5	6	9	9	6	9
Lena-Meth I	6	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Lena-Meth II	5	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Lena-Meth III	7	6	7	6	5	6	6	6	6	6	5	6	5	6	6	5
Barbara-Meth I	6	6	6	6	5	6	6	6	6	9	5	6	5	6	6	11
Barbara-Meth II	5	2	6	6	5	6	6	6	6	9	9	6	5	9	9	11
Barbara-Meth III	7	6	8	9	7	6	2	6	8	10	9	6	5	6	9	9
Baboon-Meth I	6	6	6	6	5	6	6	6	6	6	5	6	5	6	6	5
Baboon-Meth II	4	5	5	4	5	6	6	6	6	6	5	6	5	6	6	5
Baboon-Meth III	6	6	6	6	5	6	2	6	6	6	5	6	5	6	6	5

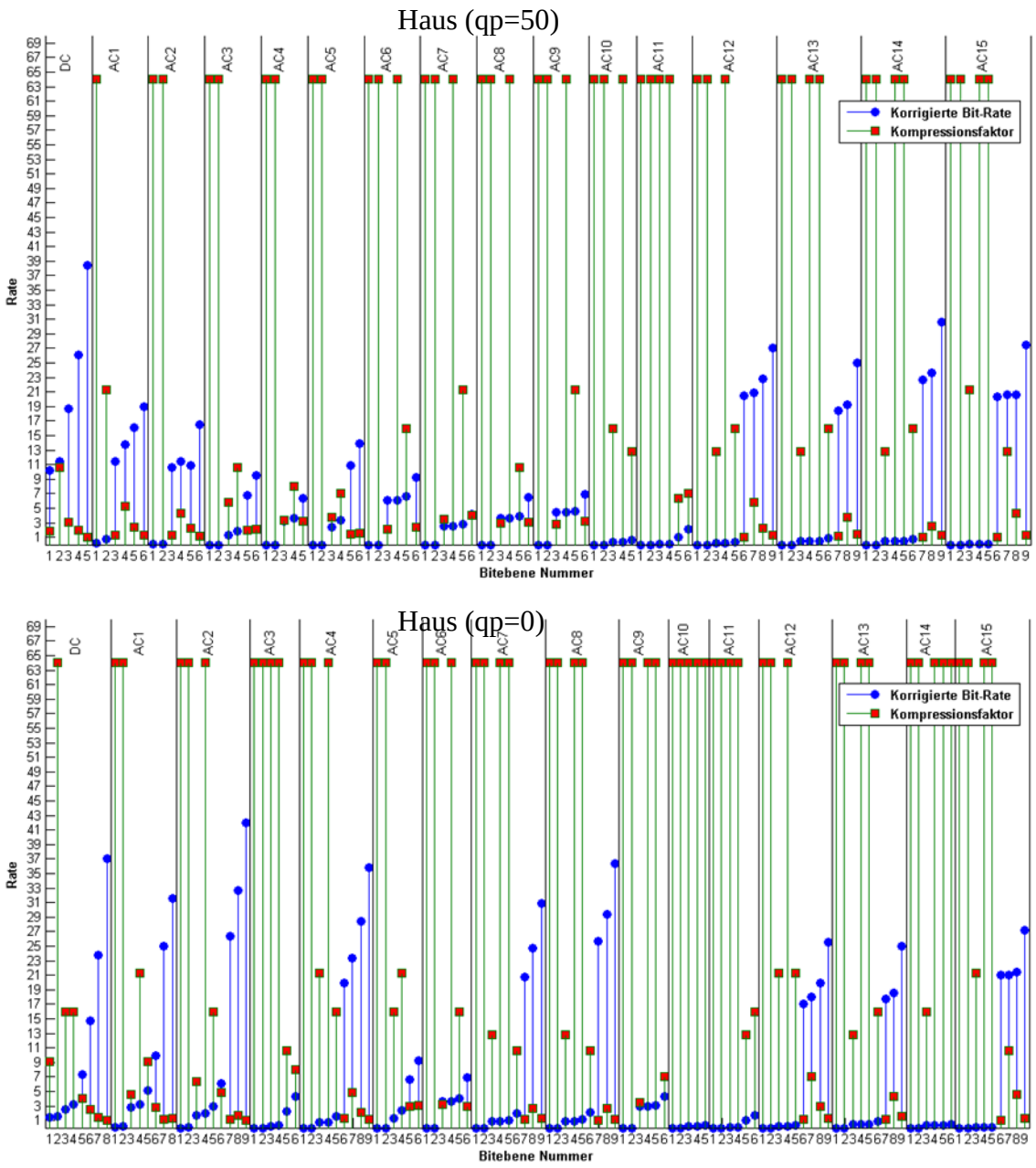


Abb.6.49: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Haus* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

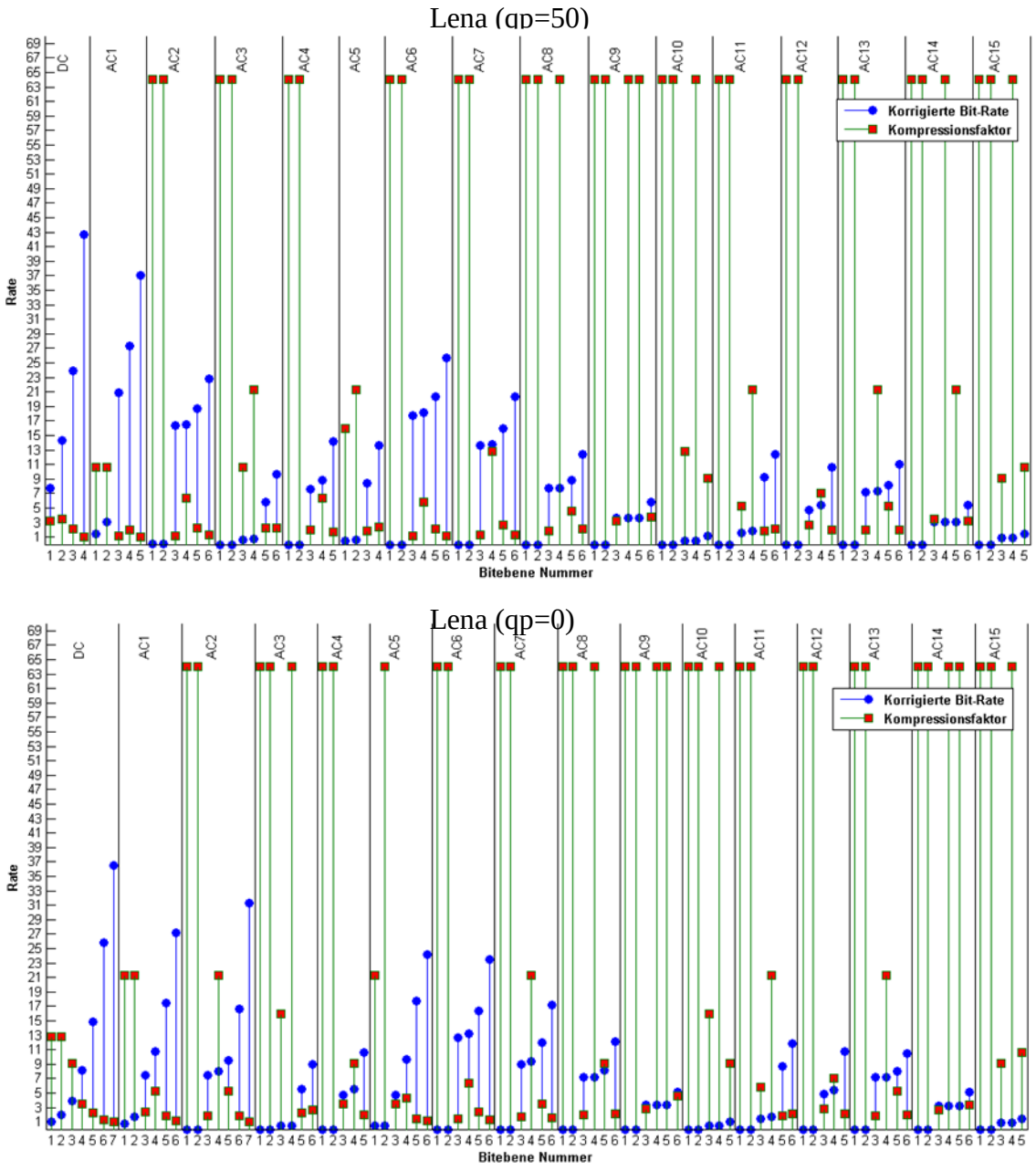


Abb.6.50: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Lena* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

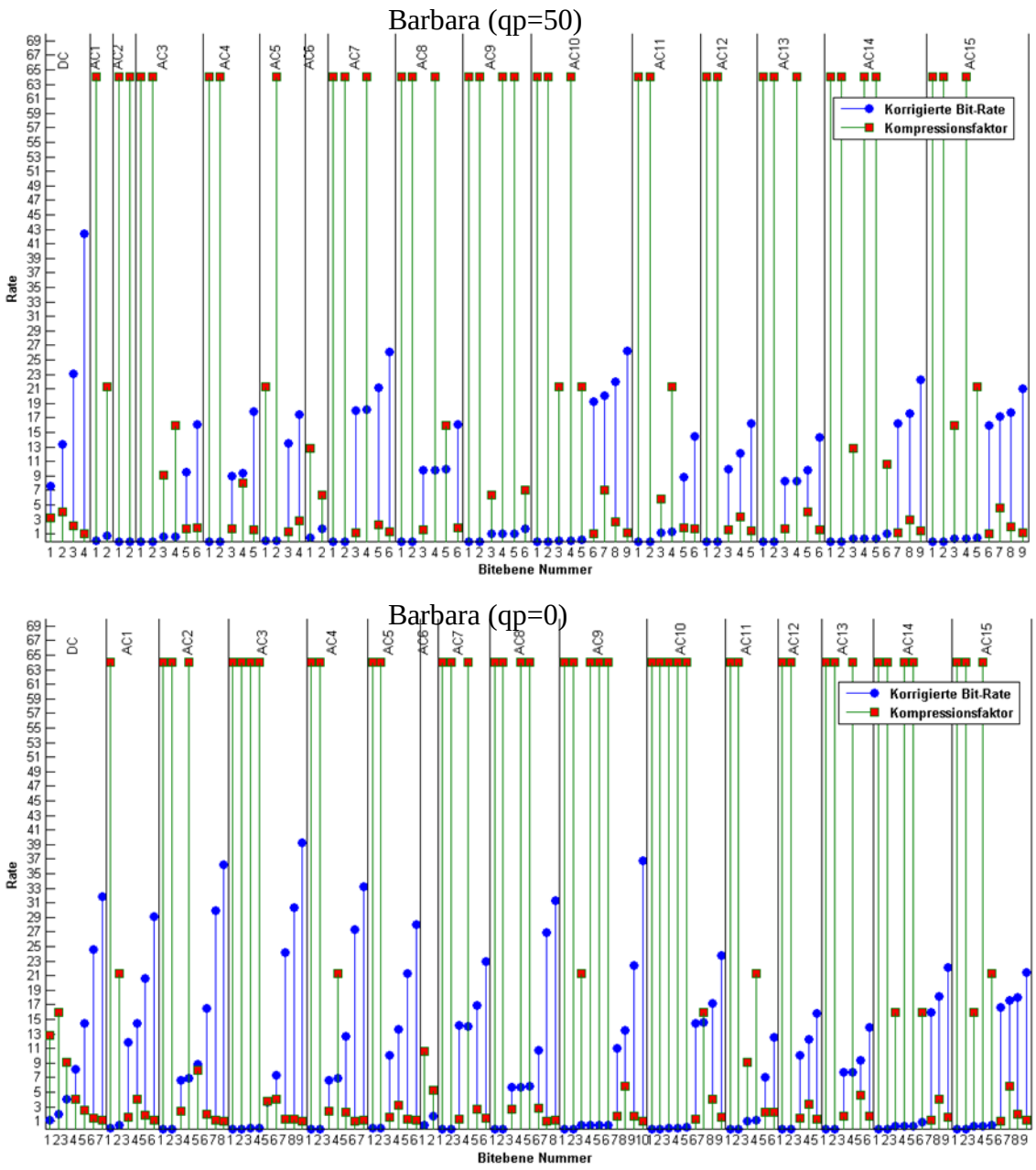


Abb.6.51: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Barbara* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0).

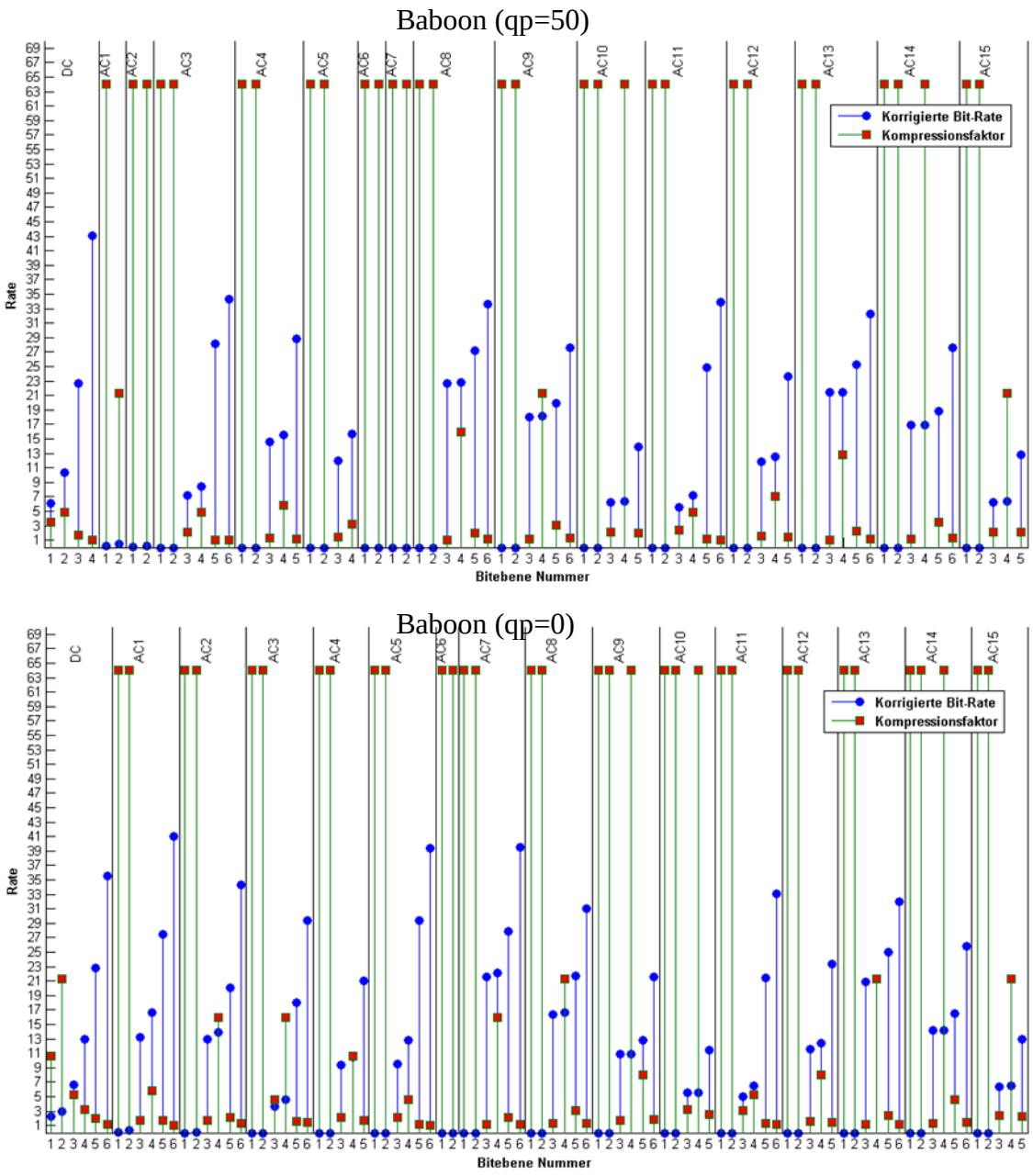


Abb. 6.52: Kompressionsfaktor und korrigierte Bit-Rate von WZ-Bild *Baboon* mit der Methode der Verteilte Bildercodierung mit linearer Pixel-Vorhersage. Die obere Abbildung zeigt den Fall, wenn die Qualität der Zusatzinformation am schlechtesten ist (Der Quantisierungsparameter des Schlüssel-Bildes qp=50). Die Abbildung unten zeigt den Fall mit der besten Qualität (qp=0)

6.4.2.3 Feedback-Kanal Rate

In der nächsten Tabelle sehen wir die Feedback-Kanal-Rate in bpp bei Anwendung der Methode III für das ganze Bild. Wir sehen wie bei Methode I und II, dass wenn der Quantizierungsparameter des Schlüssel-Bildes niedrig ist, die Anzahl der gesendeten Bits in dem Feedback-Kanal erhöht wird, weil unser Methode III mehr Bitebenen korrigiert, wenn die Zusatzinformation sehr gut ist (das ist der Fall, wenn die Qualität des Schlüssel-Bildes beim Decoder sehr gut ist). Außerdem sehen wir, dass das Bild *Haus* die geringste Feedback-Kanal-Rate zeigt; dagegen das Bild *Baboon* die höchste. Der Grund dafür ist, dass beim Bild *Haus* beim Decoder eine sehr gute Zusatzinformation vorhanden ist; beim Bild *Baboon* dagegen die schlechteste.

Im Vergleich zur Feedback-Kanal-Rate der Methode I und Methode II (siehe Tab. 6.4 und Tab. 6.10) sehen wir, dass wir in Methode III eine ein bisschen höhere Bitrate haben, weil wir in Methode III mehrere Bitebenen korrigieren können.

Tab. 6.16: Die Feedback-Kanal-Rate (bpp) der Bilder in Methode III.

Qp (Quantisierungsparameter des Schlüssel-Bildes)	Haus	Lena	Barbara	Baboon
40	0.0009	0.0015	0.0027	0.0030
35	0.0014	0.0025	0.0049	0.0051
30	0.0016	0.0027	0.0054	0.0055
25	0.0023	0.0040	0.0075	0.0072
20	0.0028	0.0056	0.0098	0.0092
15	0.0045	0.0102	0.0159	0.0142

6.4.2.4 Wahrscheinlichkeitsdichtefunktion des Korrelationsrausches der Koeffizienten

Wir haben in Abschnitt 6.2.1 (siehe Codierungsratenvorhersage in Abschnitt 6.2.1) erwähnt, dass der Decoder das Korrelationsrauschen der Koeffizienten durch das Histogramm der Differenz zwischendem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes berechnet. Dann sendet der Decoder die Varianz σ^2 dieses Histogramms an den Encoder (*Codierungsraten-Vorhersage* in Abb. 6.25).

Der Encoder nimmt an, dass die Verteilung zwischen dem Koeffizienten-Band des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes eine Laplacesche-Verteilungskurve (Laplacesche Wahrscheinlichkeitsdichtefunktion, engl:PDF) mit variance σ^2 ist.

Wir sehen in Abb.6.53 und Abb.6.54 das Histogramm (für das Bild *Haus*) der Differenz zwischen den Koeffizienten-Bänder (DC, AC1, AC2) des originalen WZ-Bildes und den Koeffizienten-Bänder des vorhergesagten WZ-Bildes, außerdem zeigen wir die Laplacesche-Verteilungskurve (PDF) für diese Koeffizienten-Bänder für Quantisierungsparameter qp=0 des Schlüssel-Bildes. Diese Histogramme wurden zum einen bei der Anwendung der Methode II und zum anderen bei der Anwendung der Methode III berechnet und alle diese geannten Abbildungen sind auf denselben Bereich normiert.

Wir ersehen aus diesen Abbildungen, dass die Laplacesche Verteilung bei Verwendung der Methode III breiter ist (größere Vorhersagefehler) als bei der Verwendung der Methode II. Dies führt zur Korrektur von wenigeren Bitebenen.

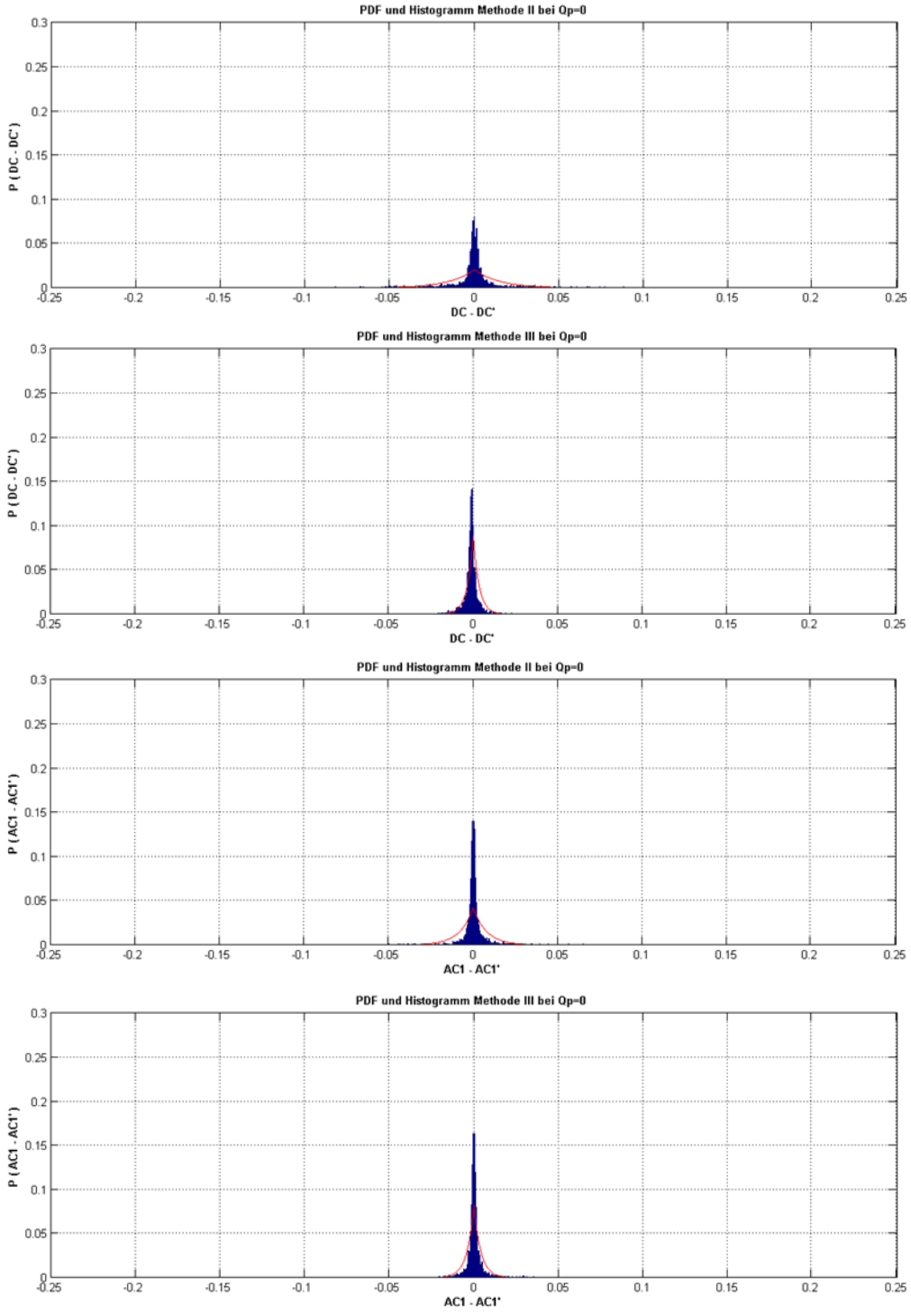


Abb.6.53: Das Histogramm der Differez zwischen der Koeffizienten-Bänder *DC,AC1* des originalen WZ-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes sowie die PDF zwischen der Koeffizienten-Bänder des decodierten Schlüssel-Bildes und der Koeffizienten-Bänder des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode III und II berechnet.

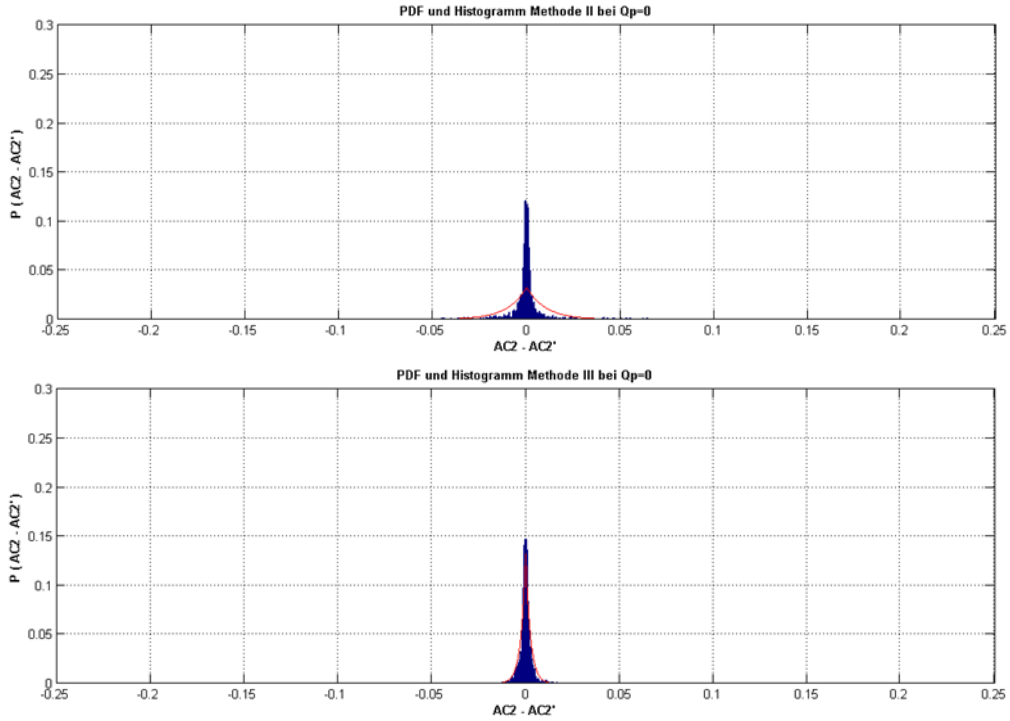


Abb.6.54: Das Histogramm der Differez zwischen dem Koeffizienten-Band *AC2* des originalen WZ-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes sowie die PDF zwischen dem Koeffizienten-Band des decodierten Schlüssel-Bildes und dem Koeffizienten-Band des vorhergesagten WZ-Bildes. Die Quantisierungsparameter des Schlüssel-Bildes sind Qp=0. Das Histogramm wurde für das Bild *Haus* in Methode III und II berechnet.

6.4.2.5 Komplexität der Encoder und der Decoder

Wir gehen von derselben Ausgangslage aus wie in Abschnitt 6.2.3.5 und benutzen dieselbe Art, um die Koplexität der Encoder und der Decoder zu berechenen.

Komplexität der Encoder:

In Tab. 6.17 und Tab. 6.18 sehen wir den Vergleich zwischen der Encoding-Zeit der Methode III und jene, wenn wir die Bilder mit H.264 Intra-Coding codieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter des Schlüssel-Bildes für Methode III, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Die Encoding-Zeit in der Methode III wie bei Methode I und II hängt besonders von dem Encoding des WZ-Bildes ab. Das heißt, wir brauchen im Vergleich zu H.264 Intra-Coding weniger Zeit mit der Methode III, weil wir die Hälfte des Bildes (das WZ-Bild) mit weniger Komplexität codieren. Das LDPC-Encoding ist viel weniger komplex als das H.264-Intra Coding.

Verglichen mit der Encoding-Zeit der Methode I (siehe Tab. 6.5 und Tab. 6.6) braucht die Methode III ein bisschen weniger Zeit, weil der Encoder in Methode II die Vorhersage-Modi für die Blöcke des WZ-Bildes im Gegensatz zur Methode I nicht berechnet. Aber die Encoding-Zeit der Methode III entspricht fast der in Methode II (siehe Tab. 6.11 und Tab. 6.12).

Tab. 6.17: Die Zeit der Codierung in Sekunden für die Bilder *Haus*, *Lena* in Methode III im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Coding in Sekunde		Methode III in Sekunde		Quotient:H.264 Intra-Coding/Methode III	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.875	1.016	0.528	0.609	1.66	1.67
35	0.968	1.203	0.580	0.716	1.67	1.68
30	1.094	1.5	0.654	0.880	1.67	1.7
25	1.297	1.907	0.778	1.116	1.67	1.71
20	1.594	2.375	0.956	1.398	1.67	1.7
15	2.078	3	1.229	1.746	1.69	1.72

Tab. 6.18: Die Zeit der Codierung in Sekunden für die Bilder *Barbara*, *Baboon* in Methode III im Vergleich zur H.264 Intra-Codierung. qp ist der Quantisierungsparameter

qp	H.264 Intra-Coding in Sekunde		Methode III in Sekunde		Quotient:H.264 Intra-Coding/Methode III	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	1.125	1.157	0.683	0.715	1.65	1.62
35	1.39	1.609	0.833	0.980	1.67	1.64
30	1.734	2.203	1.038	1.345	1.67	1.64
25	2.125	2.906	1.273	1.763	1.67	1.65
20	2.531	3.609	1.499	2.189	1.69	1.65
15	3.016	4.172	1.765	2.469	1.71	1.69

Komplexität der Decoder:

Wir werden auch hier wie bei der Codierungskomplexität die Decodierungskomplexität in Zeit messen.

In Tab. 6.19 Tab. 6.20 und sehen wir den Vergleich der Decodierung zwischen der Methode III und wenn wir die Bilder in H.264 Intra-Decoding decodieren. Der Quantisierungsparameter qp in dieser Tabelle entspricht dem Quantisierungsparameter des Schlüssel-Bildes für Methode III, und für H.264 Intra-Coding entspricht dem Quantisierungsparameter des ganzen Bildes.

Wie ersehen aus den Tabellen - wie zuvor bei der Betrachtung der Methode I und II, dass die Decodierungskomplexität der Methode III viel höher ist als die des H.264 Intra-Decodings. Wir ersehen auch aus den Tabellen, dass wenn die Quantisierung des Schlüssel-Bildes sehr fein ist (wenn qp sehr niedrig ist), wird die Decodierungskomplexität höher. Im Vergleich zur Methode II und Methode I (siehe Tab. 6.7, Tab. 6.8, Tab. 6.13 und Tab. 6.14) braucht die Methode III eine etwas höherere Decoding-Zeit, denn wie wir vorher erwähnt haben, können wir mit Methode III viele LSBs korrigeren.

Tab. 6.19: Zeit der Decodierung in Sekunde für die Bilder *Haus*, *Lena* in Methode III im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Decding in Sekunde		Methode III in Sekunde		Quotient:MethodeIII/H.264 Intra-Decoding	
	Haus	Lena	Haus	Lena	Haus	Lena
40	0.047	0.062	20.07	39.55	427.1	637.9
35	0.062	0.094	35.28	87.44	569.1	930.3
30	0.078	0.125	45.05	115.83	577.6	926.7
25	0.110	0.187	82.31	237.09	748.3	1267.9
20	0.141	0.25	120.93	407.08	857.6	1628.3
15	0.219	0.359	241.36	875.23	1102	2438

Tab. 6.20: Zeit der Decodierung in Sekunde für die Bilder *Barbara*, *Baboon* in Methode III im Vergleich zur H.264 Intra-Decodierung. qp ist der Quantisierungsparameter.

qp	H.264 Intra-Decding in Sekunde		Methode III in Sekunde		Quotient:MethodeIII/H.264 Intra-Decoding	
	Barbara	Baboon	Barbara	Baboon	Barbara	Baboon
40	0.079	0.093	81.12	156.2	1027.4	1679.3
35	0.109	0.156	172.6	385.7	1583.6	2471.3
30	0.156	0.25	275.7	663.9	1767.4	2655.3
25	0.219	0.344	494.4	1104	2257.7	3209.3
20	0.282	0.485	689.4	1668.2	2444.5	3439.3
15	0.391	0.625	1260	2513.4	3222.4	4021.3

6.4.2.6 Visueller Vergleich

Wir sehen in Abb. 6.55 das vorhergesagte Bild *Barbara* vor allen Korrekturen (bevor Paritätsbits vom Encoder gesendet wurden), codiert mit linearer Pixel-Vorhersage bei einer schlechten Qualität der Zusatzinformation (Quantisierungsparameter des Schlüssel-Bildes ist qp=40), und das gleiche Bild komplett codiert mittels H.264 Intra-Codierung bei fast gleicher Rate (bpp=0.202). In Abb. 6.56 sehen wir das gleiche Bild nach der Korrektur der Bitebenen bei gleicher Qualität der Zusatzinformation (qp=40), und wir sehen rechts das mit H.264 Intra-Codierung bei fast der gleichen Rate (bpp=1.101) komplett codierte Bild.

Allerdings weist die dritte Methode im Vergleich mit der ersten und zweiten Methode (siehe Anhangskapitel 8.4) große Flächen mit annähernd gleichem Luminanzwert auf. Dies kommt daher, dass unsere Vorhersagemethode eine pixelbasierte Interpolation darstellt. Dies kann bei geringer Quantisierung des Schlüsselbildes mit H.264 zu einer Vergrößerung der Kodierungsblöcke führen.

So sehen wir im Anhangskapitel 8.3.2, dass bei einer besseren die Qualität der Zusatzinformation (qp ist kleiner als 40), Methode III nicht mehr so sehr an Block-Artefakten und Verschwommenheit leidet.

Wir sehen im Anhangskapitel 8.3.2 , dass - wie auch bei Methode I und II - kaum noch ein wahrnehmbarer Unterschied zwischen Methode III - bei guter Qualität der Zusatzinformation - und dem H.264 Intra-Codierung besteht. Das heißt, dass bei einer sehr hohen Bitrate die Methode III für viele Anwendungung ausreichend ist.



Abb. 6.55: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=40, PSNR dieses Bildes ist 24.23 dB und die Rate 0.19 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 25.4 dB und die Rate 0.202 bpp.



Abb. 6.56: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung Verteilte Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=40, PSNR dieses Bildes ist 26.23 dB und die Rate 1.06 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 37.003 dB und die Rate 1.101 bpp.

6.4.3 Zusammenfassung

Unser Ziel in diesr Methode III war es, eine verteilte Bild Codierung zu erstellen, welche das höchtmögliche PSNR zu erreichen versucht. Wir haben die eine Hälfte des Bildes (Schlüssel-Bild) in H.264 Intra-Coding und die zweite Hälfte (WZ-Bild) mit IntDCT Transform und LDPC Encoding codiert. In der IntDCT haben wir die beste Quantisierung benutzt. LDPC-Encoding codiert den Koeffizient-Band auf Bitebene, das heißt, wir codieren zuerst die MSBs, anschließend die LSBs. Der Decoder findet bei der Verwendung des decodierten Schlüssel-Bildes und bei der Verwendung linearer Pixel-Vorhersage das vorhergesagte WZ-Bild. Der Decoder nutzt LDPC-Decoding, um das vorhergesagte WZ-Bild zu korrigieren. Wir haben also eine Hybrid-Ratenkontrolle, um wenige Feedback-Kanal-Raten zu benötigen.

Wir haben gesehen, dass das Rate-Verzerrung-Verhältnis der Methode III unter dem Rate-Verzerrung-Verhältnis der Intra-Codierung des H.264 liegt, sie aber für sehr glatte Bilder ein besseres Rate-Verzerrung-Verhältnis erzielt als mit Methode I und II. Für eine niedrige Bit-Rate (bei den meisten Bildern, wenn bpp kleiner ist als 0.2) konnten wir fast das gleiche PSNR erreichen wie bei H.264, und bei dieser niedrigen Bit-Rate konnten wir nicht so viele Bitebenen korrigieren (keine Verbesserung in PSNR durch Senden eines längeren Paritätsvektoren): Wir haben auch herausgefunden, dass wir bei sehr hoher Bit-Rate die PSNR stark verbessern können. Der wahrnehmbare Unterschied bei sehr hoher Bit-Rate zwischen der Methode III und dem H.264 Intra-Coding nicht groß ist. Deswegen lohnt es sich, für manche Anwendungen die Methode III statt H.264 Intra-Coding zu benutzen.

Die Codierungskomplexität der Methode III ist geringer ist als die Komplexität des H.264 Intra-Coding sowie der Methode I, aber sie ist fast wie die der Methode II. Die Decodierungskomplexität ist sehr viel höher als die des H.264 Intra-Decoding und auch als die der Methode I und II. Wir haben auch gesehen, dass die Feedback-Kanal-Rate wie bei Methode III sehr gering ist und etwas höher liegt als Methode I und II.

6.5 Vergleich der Arbeit mit anderen Papers der verteilten Bildcodierung

In diesem Abschnitt gibt es einen Überblick über state-of-the-art Artikel über verteilte Bildcodierung, und ich vergleiche meine Arbeit mit diesen anderen Arbeiten.

Das Prinzip verteilter Bildcodierung ist, die Information des Bildes in zwei oder mehrere Teile aufzuteilen und dann jeden Teil an einem eigenen Encoder zu codieren. Die Voraussetzung und Herausforderung bei verteilter Quellencodierung ist, dass die Encoder nicht miteinander kommunizieren dürfen. Um die Arbeit mit den anderen vergleichen zu können, müssen wir nicht nur die Rate-Distortion-Kurven sondern auch die Encoder und die Decoderkomplexität vergleichen. Der Grund dafür ist, dass man bei verteilter Bildcodierung bessere Rate-Distortion-Kurven bekommen kann, wenn ein großer Teil der Information mit einem guten Encoder (wie JPEG oder H.264 Intra Coding) kodiert wird und nur ein kleinerer Teil mit einem einfachen Encoder (wie zum Beispiel nur DCT Transformation). In diesem genannten Beispiel erreicht man gute Rate-Distortion-Kurven, aber die Gesamtkomplexität der Encoder wird größer sein als wenn man den kleineren Teil der Information mit guten Encodern und den großen Teil mit einem einfachen Encoder kodiert. Normalerweise bezeichnen wir die Information, die mit einem einfachen Encoder kodiert wird als WynerZiv-Anteil und die andere Information, die mit den komplexeren Encodern kodiert wird, als Key-Anteil. Die Seiteninformation am Decoder ist nichts anderes als der vorhergesagte Wyner-Ziv-Anteil.

Die wichtigste Arbeit über verteilte Bildcodierung ist [ZLC08]. Die Autoren designen einen Codec in der pixel-domain, um räumliche Redundanz auszunutzen und schlagen ein Markov Random Field-Modell (MRF) am Decoder vor, um räumliche Korrelation zu repräsentieren. Sie haben dabei dieses Modell am Decoder benutzt, um die Wahrscheinlichkeit der zu decodierenden Bits am LDPC-Decoder zu verändern, und gleichzeitig wurden die korrigierten Bits am Ausgang des LDPC-Decoders benutzt, um das MRF des Bildes zu verbessern. In diesem Paper wird LDPC auf der Bitebene der Pixel eingesetzt, und der LDPC-Decoder und das MRF tauschen iterativ Informationen aus, bis der Algorithmus konvergiert. Immer, wenn das MRF-Modell Informationen vom LDPC-Decoder erhält, wird das Bild geschätzt, und ein MRF wird aufgebaut, um die bedingte Wahrscheinlichkeit für jeden Pixel in Bezug auf seine Nachbarn zu bestimmen. Je höher die räumliche Korrelation des Bildes ist, umso stärker ist der Einfluss des MRF auf das LDPC-Decodieren. Es ist anzumerken, dass der Encoder in dieser Arbeit LDPC benutzt, um die Pixelwerte zu encodieren. Der Nachteil dieser Arbeit ist, dass die Autoren voraussetzen, dass die gemeinsame Verteilung (das statistische Verhältnis zwischen Seiteninformation und den zu codierenden Informationen) am Decoder vorhanden ist. Diese Annahme ist in der Realität im Allgemeinen nicht gegeben. In meiner Arbeit habe ich das nicht so angenommen, sondern ich setze voraus, dass weder Encoder noch Decoder die Statistiken zwischen Seiteninformation (den vorhergesagten Wyner-Ziv-Anteil am Decoder) und dem ursprünglichen Anteil kennen. In meiner Arbeit wird der decodierte “Key part” und die Seiteninformation verwendet, um diese Statistiken vorherzusagen. Der zweite Nachteil in dem genannten Paper ist, dass die Autoren angenommen haben, dass ein verrauschtes Bild bereits am Decoder vorhanden ist. Wegen dieser Annahme haben sie die Rate-

Distortion-Kurve (PSNR vs. Bit pro Pixel) gezeigt, ohne dabei die Kosten (bpp) mit einzurechnen, um das verrauschte Bild zum Decoder zu senden. Daher ist die Rate-Distortion-Kurve bei ihrem Paper besser als sie wäre, würde man das Bild mit JPEG kodieren. Es fehlt in diesem Paper die Information über die Komplexität der Encoder und der Decoder. Diese wurde nicht berechnet, sondern es wurde nur gesagt, dass die Komplexität der Encoder geringer ist als bei JPEG. Wegen dieser fehlenden Informationen kann ich die Komplexität der Encoder und Decoder in meiner Arbeit und in diesem Paper nicht vergleichen. Die Rate-Distortion-Kurven sind wegen der Annahme zum am Decoder vorhandenen verrauschten Bild ebenfalls nicht vergleichbar. Im Grunde genommen ist jeder Vergleich zwischen diesem Paper und meiner Arbeit bereits aufgrund der unterschiedlichen Grundannahmen schwierig.

Eine weitere Arbeit über verteilte Bildcodierung ist [CDB05]. In dieser Arbeit nutzen die Autoren verteilte Bildcodierung nur, um den Fehler an Kanten im Bild zu korrigieren. Sie verwenden nicht die wirklichen Pixelwerte zur Codierung der Seiteninformationen. Stattdessen benutzen sie die niederfrequenten Anteile des Bildes. Außerdem verwenden sie Nebenklassen variierbarer Länge (engl:variable-length coset), um den Peak-SNR an Bildkanten zu erhöhen. Die Autoren nehmen in ihrer Arbeit an, dass die gemeinsame Verteilung, die den Eingangsdaten zugrunde liegt, am Encoder bekannt ist, um die Encoder-Bitrate zu bestimmen. Diese Annahme ist in der Realität nicht immer gegeben. Das vorgeschlagene Decodierschema enthält einen iterativen Prozess, um die Seiteninformation zu rekonstruieren, indem Wavelet-Zerlegung zusammen mit der korrekten Information der Nebenklasse (engl:coset) verwendet wird. Der Nachteil der Verwendung der Wavelets ist jedoch, dass die Wavelet-Transformation komplexer ist als die DCT. Daher ist der Encoder hier komplexer aufgebaut. Die Autoren in diesem Paper haben nicht die Komplexität der Encoder und der Decoder in ihrem Paper gezeigt. Ich kann meine Arbeit auch deshalb nicht mit diesem Paper vergleichen, weil die Autoren die verteilte Bildcodierung nur einsetzen, um die Kanten des Bildes zu verbessern.

Die Autoren des Artikels [Liv02] zeigen ein verteiltes Quellencodierungsschema für korrelierte Bilder, das Modulo-Codierung von Pixelwerten sowie binäre und nichtbinäre Turbocodes nutzt, so dass höhere Bitrateneinsparungen als nur mit Modulo-Codierung praktisch verlustfrei erreicht werden können. Die Komplexität der Methode wurde nicht angegeben. Der Nachteil in dieser Arbeit ist, dass das ganze Bild am Eingang eines der beiden Encoder verfügbar sein muss. Dies ist ein Widerspruch zur Grundannahme der verteilten Codierung. Außerdem haben die Autoren sich nur mit dem Fall Gaußschen Rauschens beschäftigt, und sie haben angenommen, dass am Decoder dieses Rauschmodell bekannt ist. Daher ist dieses Modell nicht für alle Bilder geeignet, sondern nur wenn das Rauschmodell korrekt ist. Meine Arbeit funktioniert für alle Bilder, egal welches Rauschen auftritt, weil am Decoder das Rauschen geschätzt wird und der Feedback-Kanal genutzt wird, um diese Schätzung zu verbessern. Wegen dieser unterschiedlichen Annahmen in diesem Paper und meiner Arbeit ist es nicht korrekt, die

entstandenen Rate-Distortion-Kurven zu vergleichen, und aufgrund fehlender Informationen lässt sich die Komplexität ebenfalls nicht vergleichen.

Im Artikel [WL09] schlagen die Autoren ein DCT-basiertes verteiltes Quellencodierungsschema niedriger Komplexität vor, um hyperspektrale Bilder zu komprimieren. Sie nutzen einen Ansatz, der auf der Zerlegung von Mengen von DCT-Koeffizienten basiert, um dann Bitebenen zu extrahieren; Danach wenden sie einen Slepian-Wolf Coder an, um die verteilte Bildcodierung durchzuführen. Ihre Arbeit beruht darauf, dass die Abhängigkeit von zwei zu encodierenden Teilinformationen in einem linearen Filter erster Ordnung dargestellt werden kann. Diese Annahme passt oft nur für Hyperspektralbilder. In ihren Ergebnissen zeigen sie, dass das vorgeschlagene Modell um bis zu 0.4 dB bessere Codierungswerte erreichen kann als verteilte Quellencodierung im diskreten Wavelet-Transformationsbereich, aber sie haben nicht die Komplexität der Methode angegeben. Diese Arbeit passt nur, wenn die Korrelation zwischen der Information an den Encodern linear ist. Daher passt diese Methode nur für hyperspektrale Bilder, aber meine Arbeit ist für alle Arten von Bildern geeignet. Außerdem kommunizieren die Encoder in diesem Paper miteinander, und an dieser Stelle trifft die Annahme bei der verteilten Quellencodierung nicht zu. Daher sind die Ergebnisse meiner Arbeit mit denen dieses Papers nicht vergleichbar.

Das Schema im Artikel [LVG09] verwendet einen Expectation Maximization Algorithmus in einem Slepian-Wolf-Decoder. Die Authentifizierung in ihrer Arbeit erfordert eine Slepian-Wolf-codierte Bildprojektion, die dem Decoder bekannt ist. Sie erweitern ihr Schema, um Bilder zu authentifizieren, die in den Bereichen Kontrast, Helligkeit und affinem Warping verändert wurden. Diese Arbeit beschäftigt sich nur mit der Authentifizierung der Bilder und hat deshalb einen völlig anderen Fokus als meine Arbeit.

Der Artikel [YDD07] schlägt eine Methode zur Codierung von Multi-view Bildern vor, die sich auf "Multiterminal source coding" (MSC) stützt. Dank der separaten Codierung in MSC kann das vorgestellte Codierschema gute „Random access performance" erzielen, und die räumliche Redundanz kann selbst dann ausgenutzt werden, wenn die Encoder nicht miteinander kommunizieren können. Das Codierschema kann einfach auf N Ansichten erweitert werden, und die Kompressionsrate jeder Ansicht kann leicht angepasst werden, um den aktuellen Ressourcenanforderungen wie Bandbreite oder Speicher zu genügen. Die Ergebnisse des Experiments zeigen, dass die Kompressionsleistung besser ist als die des JPEG-En- und Decoders. Der Nachteil dieser Arbeit ist, dass die Autoren voraussetzen, dass die gemeinsame Verteilung, die den Eingangsdaten zugrunde liegt, am Decoder vorhanden ist. Diese Annahme trifft aber in der Realität im Allgemeinen nicht zu. Diese Arbeit beschäftigt sich nur mit Multi-view-Bildern und hat deshalb eine völlig andere Anwendung als meine Arbeit.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

In dieser Arbeit werden insgesamt drei praktische verteilte Bildercodierungen vorgestellt. Bei jeder Methode teilen wir das Bild zu zwei Teile auf. Der Erste Teil des Bildes (Wyner-Ziv-Bild) wird mittels der ganzzahligen Diskreten Cosinus-Transformation (engl: integer discrete cosine transform = DCT) codiert, anschließend wird die Slepian-Wolf-Codierung (Kanalcodierung) angewandt. Hierbei wird die Methode der "ASWC (asymmetrische Slepian-Wolf-Codierung) mit der Paritätsmethode" eingesetzt. Es werden also ausschließlich Paritätsbits erzeugt und über einen Kanal versendet. Zur Konstruktion der Paritätsbits wird der Niedrigdichten-Paritäs-Prüfcode (engl: low density parity check code = LDPC) benutzt.

Der zweite Teil des Bildes ist Schlüssel-Bild. Das herkömmlich codierte Schlüssel-Bild wird bei der Decodierung verwendet um eine Schätzung des Wyner-Ziv-Bildes durchzuführen, welches mit Hilfe des Kanalcodes verfeinert wird. Der Vorteil dieses Ansatzes gegenüber bekannten Bildcodierungsverfahren wie JPEG, JPEG2000 oder auch H.264 Intra-Codierung besteht vor allem in der Verschiebung der Codierungskomplexität für das Wyner-Ziv-Bild vom Encoder in den Decoder.

Die Decodierung des LDPC ist die weiche LDPC-Decodierung (engl:soft LDPC-Decoding), und wir gehen in unserer Arbeit davon aus, dass die Quellen (Wyner-Ziv-Bild und Schlüssel-Bild) binär sind und ihre Wahrscheinlichkeitsverteilungsfunktionen gleich verteilte Funktionen sind.

Bei allen unseren Methoden nutzten wir die Hybrid-Ratenkontrolle zwischen dem Encoder und dem Decoder. Deren Vorteil liegt darin, dass sie die Komplexität der Slepian-Wolf-Decodierung (insbesondere der LDPC-Decodierung) reduziert, weil die Fehlerfrei-Decodierung besser geschieht als bei einer reinen Decoding-Ratenkontrolle.

Das virtuelle Kanal-Modell zwischen Quelle und Zusatzinforation (das vorhergesagte WZ-Bild) ist in dieser Arbeit der BSC-Rauschkanal und die Rausch-Verteilungsfunktion ist die Laplacesche Verteilung mit dem Mittelwert Null.

Wir haben festgestellt, dass nur bei einem sehr glatten Bild (d.h. bei hoher Intra-Korrelation) die Methode III "verteilte Bildercodierung mit linearer Pixel-Vorhersage" die beste Methode (bezüglich des PSNR) ist. Für die meisten herkömmlichen Bilder (außer für sehr glatte Bilder) hat sich die Methode I "verteilte Bildercodierung mit Intra-Vorhersage-Modi" als die beste PSNR erwiesen, jedoch ist bei dieser Methode die Codierung etwas komplizierter als bei den anderen Methoden. Wir haben auch gesehen, dass Methode II im Vergleich zur Methode I eine niedrigere PSNR erzielt.

Für alle unsere Methoden gilt, dass wenn die Qualität der Zusatzinformation sehr niedrig ist, wir annähernd die gleiche Verzerrung wie mit der H.264 Intra-Codierung erzielen können, aber wir können nicht viele Vorhersagefehler des Wyner-Ziv-Bildes korrigieren. Daher ist es in diesem Fall besser, ohne jede Korrektur das vorhergesagte Wyner-Ziv-Bild zu erzielen (ohne Bits vom Wyner-Ziv-Encoder an den Decoder zu senden). Wenn die Qualität der Zusatzinformation sehr hoch ist, haben wir kaum einen wahrnehmbaren Qualitätsunterschied zwischen unseren Methoden und der H.264 Intra-Codierung feststellen können. Daraus folgt, dass es in diesem Fall für viele Anwendungen besser ist, die verteilte Bildercodierung anstatt der Intra-Codierung zu nutzen, da bei verteilter Bildercodierung ein Teil der Codierungskomplexität an den Decoder verlagert wird.

Die Codierungskomplexität aller unserer Methoden ist geringer als die der H.264 Intra-Codierung (die Reduzierung der Komplexität ist von bis zu 42% im Encoder– verglichen mit dem H.264 Intra-Encoder), aber die die Decodierungskomplexität ist bei ihnen viel höher.

In allen unseren Methoden haben wir einen Feedback-Kanal benutzt und dabei festgestellt, dass die Feedback-Kanal-Rate vernachlässigbar ist.

7.2 Ausblick

In diesem Kapitel stellen wir einige Richtungen vor, in denen unsere Arbeit weitergeführt werden könnte.

1) **Änderung der Quantisierung in Abhängigkeit von den Frequenz-Transformationskoeffizienten und von der Änderung des Modells des virtuellen Kanals:**

   Aus dem Kompressionsfaktor der Bitebenen konnten wir ersehen, dass weniger Bits gesendet werden können, wenn die Frequenz-Transformationskoeffizienten unterschiedlich quantisiert werden. In anderen Worten sollte für die Hochfrequenz-Transformationskoeffizienten eine nicht-feine Quantisierung vorgesehen werden, für die Niedrigfrequenz-Transformationskoeffizienten hingegen eine feine Quantisierung.

   Es würde auch eine Verbesserung bringen, den Quantisierungsprozess am Encoder in Abhängigkeit von den Veränderungen des Modells für den virtuellen Kanal zwischen den Quelleninformationen (die zu codierenden Pixel oder IntDCT-Koeffizienten) und der Zusatzinformation zu ändern.

2) Anpassungsfähige Blockgröße:

In unserer Arbeit haben wir die ganzzahlige DCT nur für 4×4 Blöcke angewendet. Aber wenn wir in einem großen Teil des Bildes (größer als 4×4 Pixel) eine geringe Intra-Korrelation haben, wird es besser sein, in diesem Teil die ganzzahlige DCT anzuwenden, ohne diesen in kleinere Blöcke aufzuteilen. Wir können eine Wyner-Ziv- Codierung konstruieren, die dies berücksichtigt, dazu prüft diese Codierung die wichtigsten Statistiken des Wyner-Ziv-Bildes, bevor der eigentliche Codierungsprozess des Bildes beginnt. In diesem Szenario muss der WZ-Encoder die Positionen und Größen aller Blöcke im WZ-Bild an den Decoder senden.

3) Änderung der Glättungsbedingungen in Methode II (verteilte Bildercodierung mit maximal glatter Bild-Wiederherstellung):

In Methode II haben wir die Glättungsbedingungen als einen konstanten Wert eingesetzt. Aber wenn wir diese Bedingungen in Abhängigkeit von den umgebenden Pixeln aus dem Schlüssel-Bild ändern, wird dies zu einer besseren Vorhersage für das Wyner-Ziv-Bild führen. Dies hieße des Weiteren, dass die Statistiken des Schlüssel-Bildes am Decoder berechnet und die Glättungsbedingungen entsprechend gewählt werden müssen.

4) Modell des virtuellen Kanals:

In unserer Arbeit haben wir das BSC-Kanalmodell für den virtuellen Kanal zwischen dem Wyner-Ziv-Bild und der Zusatzinformation genutzt. Die Wahrscheinlichkeitsverteilungsfunktion des Rauschens in diesem Kanal war eine Laplacesche Verteilungsfunktion mit dem Mittelwert Null.

Wir sollten anmerken, dass dieses Modell des virtuellen Kanals nicht immer zutrifft, insbesondere wenn die Korrelation zwischen Zusatzinformation und dem WZ-Bild sehr niedrig ist. Wir sollten auch anmerken, dass selbst in dem Fall, wenn diese Korrelation hoch ist, die Korrelation zwischen einigen Teilen der Zusatzinformation (Pixel oder IntDCT-Koeffizienten) und deren Entsprechung im WZ-Bild durch unser Modell des virtuellen Kanals eventuell nicht gut dargestellt wird. Solch eine Situation liegt bei den Wyner-Ziv-Bildblöcken am Rande des Originalbildes vor, da die Vorhersage für diese Blöcke am Decoder schlechter ist als für die anderen Wyner-Ziv-Bildblöcke. So können wir sagen, dass die Korrelation zwischen der Quelle (WZ-Bild) und der Zusatzinformation räumlich variiert, so dass ein nicht-stationärer virtueller Korrelationskanal zwischen der Quelle und der Zusatzinformation entsteht.

Folgend aus all dem schlagen wir vor, nicht nur ein virtuelles Kanalmodell zu benutzen, sondern mehrere davon, und zwar in Abhängigkeit zur Statistik des WZ-Bildes. Zu diesem Thema gibt es einige Papers wie [QHL07], [WGL05].

8 Anhänge

8.1 Visueller Vergleich zwischen Methode I und H.264 Intra-Coding

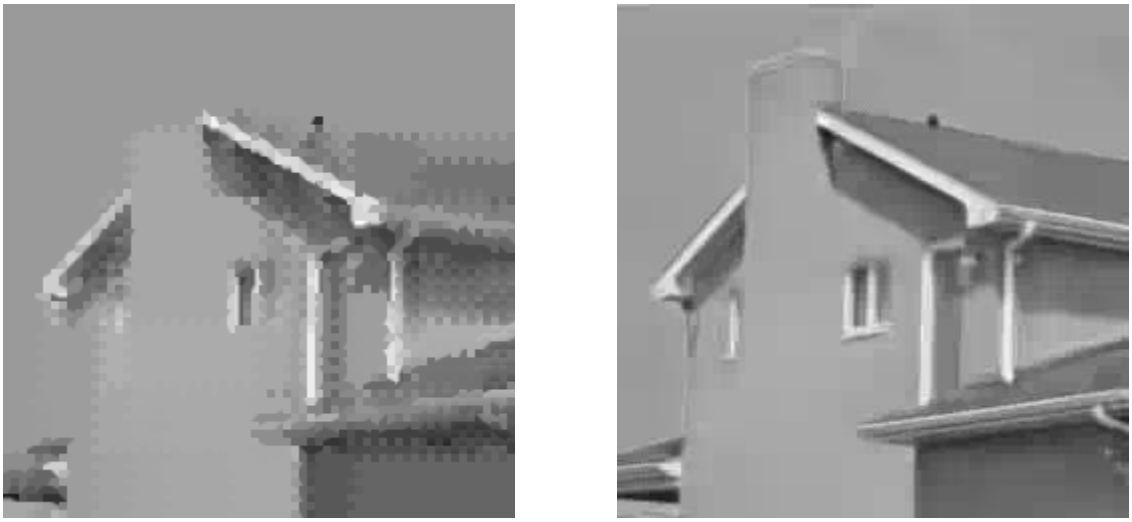


Abb. 8.1: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 26.76 dB und die Rate 0.18 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 35.859dB und die Rate 0.183bpp.

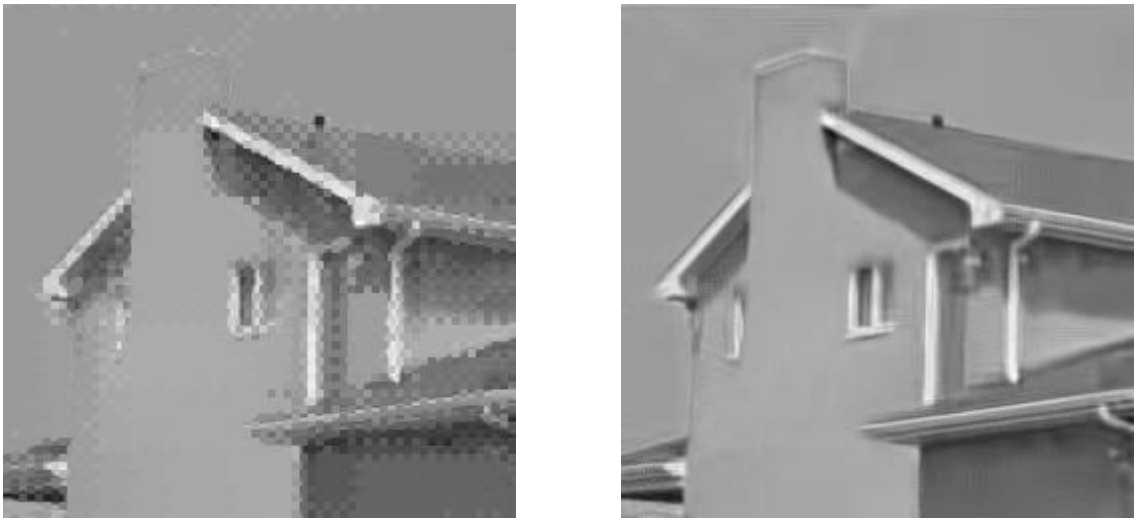


Abb. 8.2: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 28.05 dB und die Rate 0.73bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp.



Abb. 8.3: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 35.83dB und die Rate 0.36 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=27), PSNR diese Bildes ist 40.02 dB und die Rate 0.373 bpp.



Abb. 8.4: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 38.55 dB und die Rate 1.03 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=18), PSNR diese Bildes ist 45.52 dB und die Rate 1.013 bpp.





Abb. 8.5: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 28.03 dB und die Rate 0.31 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=37), PSNR diese Bildes ist 30.619 dB und die Rate 0.329 bpp.





Abb. 8.6: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 30.15 dB und die Rate 0.86 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 36.915 dB und die Rate 0.829 bpp.





Abb. 8.7: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi, Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 31.7 dB und die Rate 0.83 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 36.915 dB und die Rate 0.829 bpp.





Abb. 8.8: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 36.23 dB und die Rate 1.32 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=23), PSNR diese Bildes ist 40.612 dB und die Rate 1.277 bpp.



Abb. 8.9: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 29.53 dB und die Rate 0.96 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.035 dB und die Rate 1.001 bpp.



Abb. 8.10: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 35.41dB und die Rate 1.67 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=22), PSNR diese Bildes ist 42.092 dB und die Rate 1.715 bpp.



Abb. 8.11: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 21.21 dB und die Rate 0.18 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=44), PSNR diese Bildes ist 22.722 dB und die Rate 0.167 bpp.



Abb. 8.12: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 22.65 dB und die Rate 1.02 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 29.52 dB und die Rate 1.009 bpp.

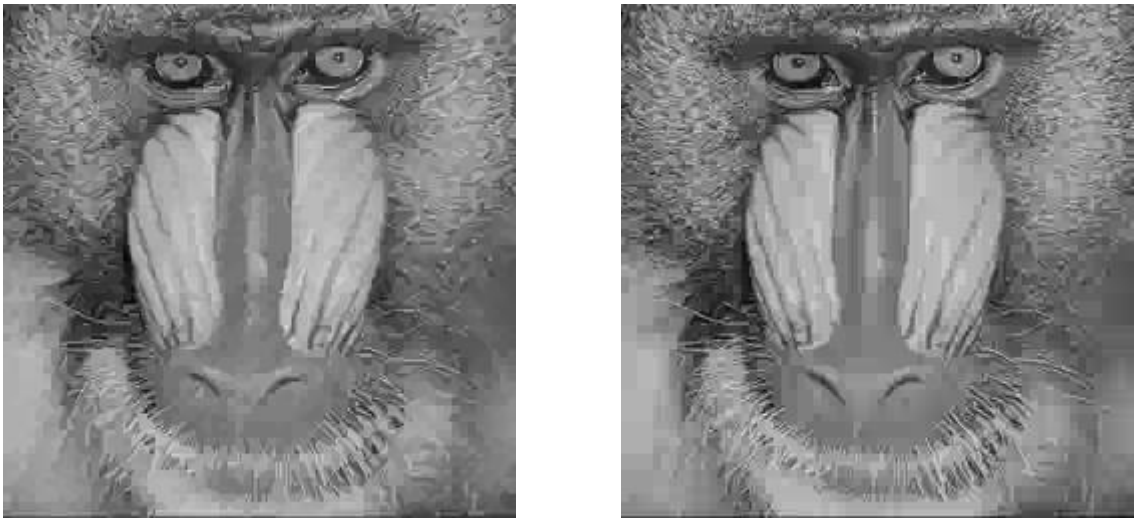


Abb. 8.13: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung Verteilte Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 24.25 dB und die Rate 0.38 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=40), PSNR diese Bildes ist 24.976 dB und die Rate 0.396 bpp.

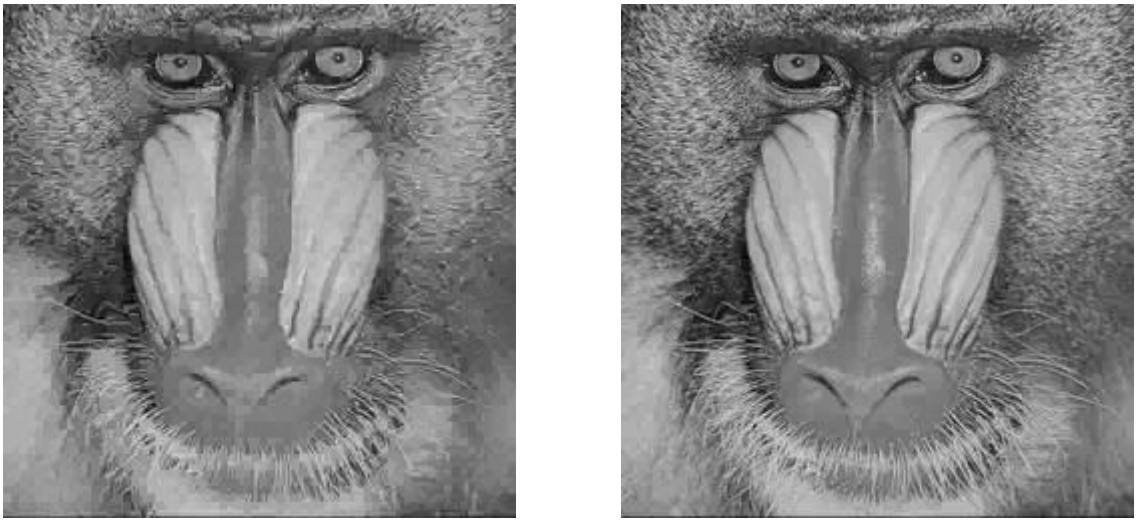


Abb. 8.14: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit Intra-Vorhersage-Modi. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 26.34 dB und die Rate 1.22 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=32), PSNR diese Bildes ist 31.104 dB und die Rate 1.259 bpp.

8.2 Visueller Vergleich zwischen Methode II und H.264 Intra-Coding

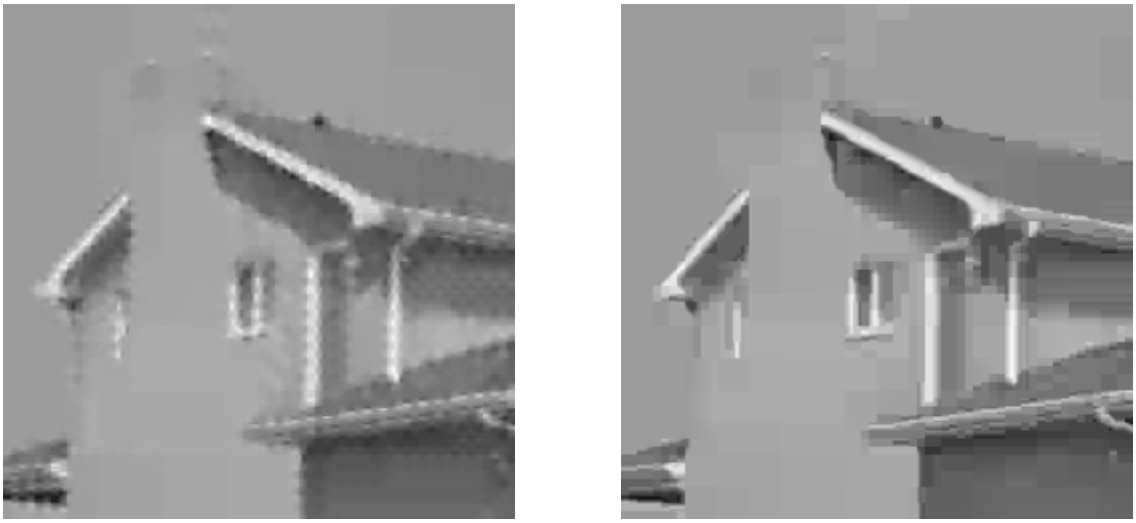


Abb. 8.15: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung, Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 29.72 dB und die Rate 0.08 bpp. Das rechte Bild wurde komplett mit H.264codiert (qp=43), PSNR diese Bildes ist 30.86 dB und die Rate 0.08 bpp.



Abb. 8.16: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 33.18 dB und die Rate 0.75 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp.

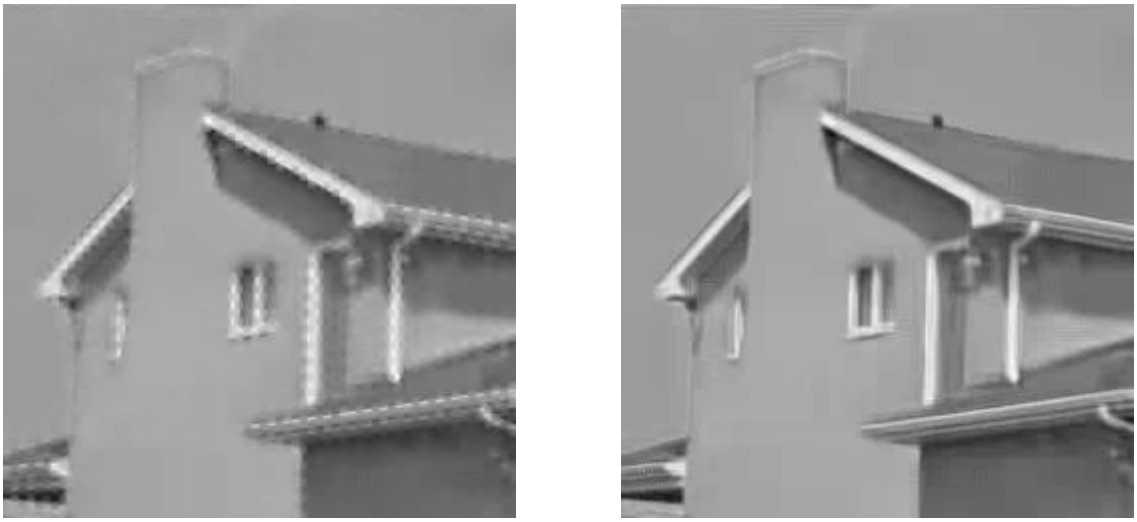


Abb. 8.17: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 32.54 dB und die Rate 0.34 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 39.506 dB und die Rate 0.34 bpp.



Abb. 8.18: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=25. PSNR dieses Bildes ist 39.16 dB und die Rate 1.08 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=18), PSNR diese Bildes ist 45.520 dB und die Rate 1.013 bpp.



Abb. 8.19: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 25.62dB und die Rate 0.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 26.727 dB und die Rate 0.159 bpp.



Abb. 8.20: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 29.04 dB und die Rate 0.76 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.073 dB und die Rate 0.746 bpp.



Abb. 8.21: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 27.42 dB und die Rate 0.44 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 32.607 dB und die Rate 0.451 bpp.



Abb. 8.22: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 33.48 dB und die Rate 1.04 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=26), PSNR diese Bildes ist 38.373 dB und die Rate 0.988 bpp.



Abb. 8.23: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=0. PSNR dieses Bildes ist 26.08 dB und die Rate 2.89 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=13), PSNR diese Bildes ist 49.923dB und die Rate 2.936 bpp.



Abb. 8.24: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=0. PSNR dieses Bildes ist 30.89 dB und die Rate 3.82 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=8), PSNR diese Bildes ist 54.117 dB und die Rate 3.76 bpp.



Abb. 8.25: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 20.59 dB und die Rate 0.03 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=51), PSNR diese Bildes ist 20.778 dB und die Rate 0.034 bpp.



Abb. 8.26: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=50. PSNR dieses Bildes ist 22.1 dB und die Rate 0.80 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=36), PSNR diese Bildes ist 27.82 dB und die Rate 0.765 bpp.

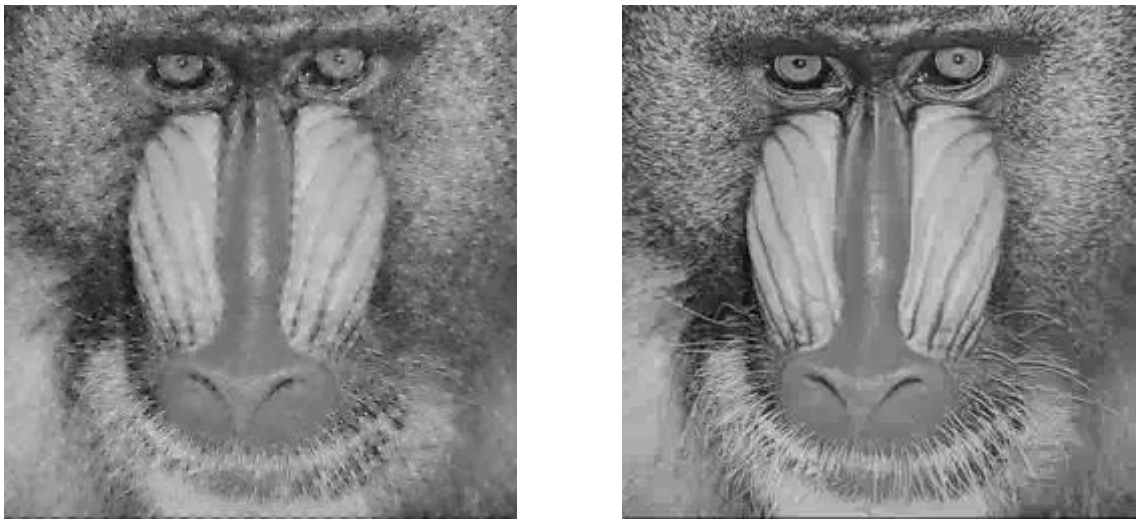


Abb. 8.27: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 24.66 dB und die Rate 0.86 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=35), PSNR diese Bildes ist 28.605 dB und die Rate 0.874 bpp

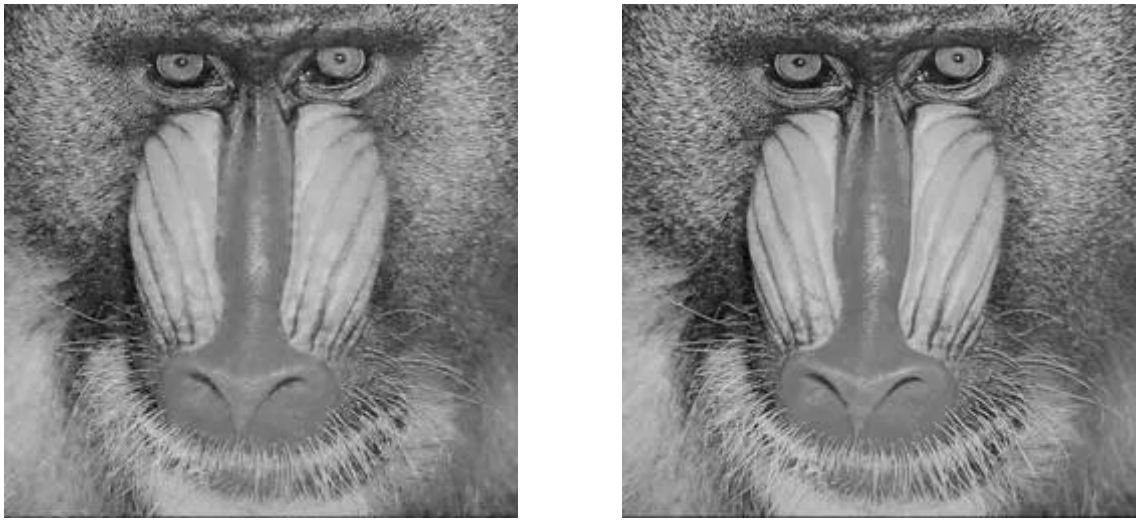


Abb. 8.28: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit maximal glatter Bild-Wiederherstellung. Der Quantisierungsparameter des Schlüssel-Bildes qp=30. PSNR dieses Bildes ist 28.66 dB und die Rate 1.71 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 33.699 dB und die Rate 1.7 bpp.

8.3 Original Bilder und Visueller Vergleich zwischen Methode III und Intra-Coding

8.3.1 Original WZ-Bilder und Schlüßel-Bilder

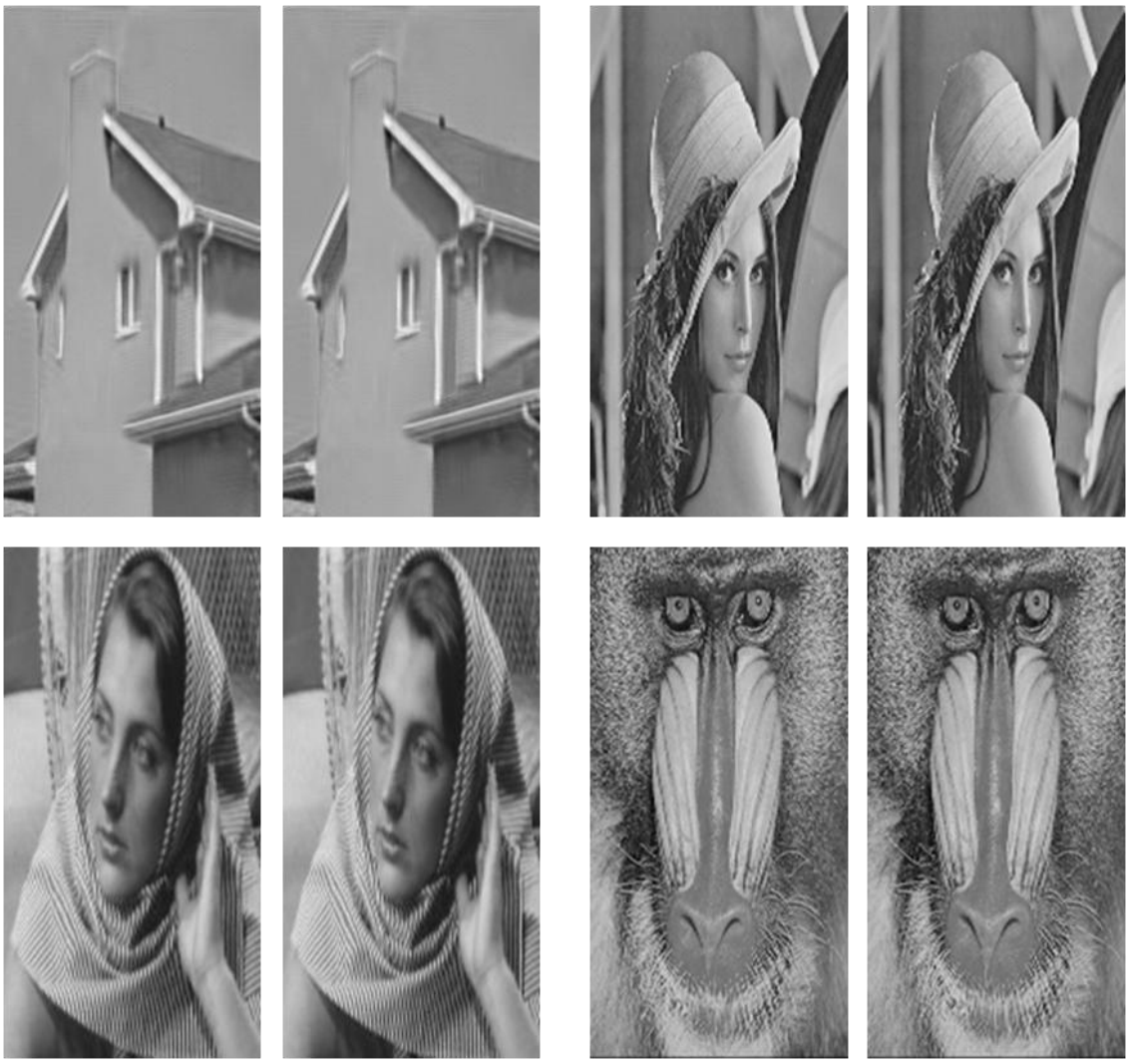


Abb. 8.29: Die Original-WZ-Bilder rechts und die Original-Schlüssel-Bilder links, in Methode III (verteilte Bildercodierung mit linearer Pixel-Vorhersage).

8.3.2 Visueller Vergleich zwischen Methode III und H.264 Intra-Coding

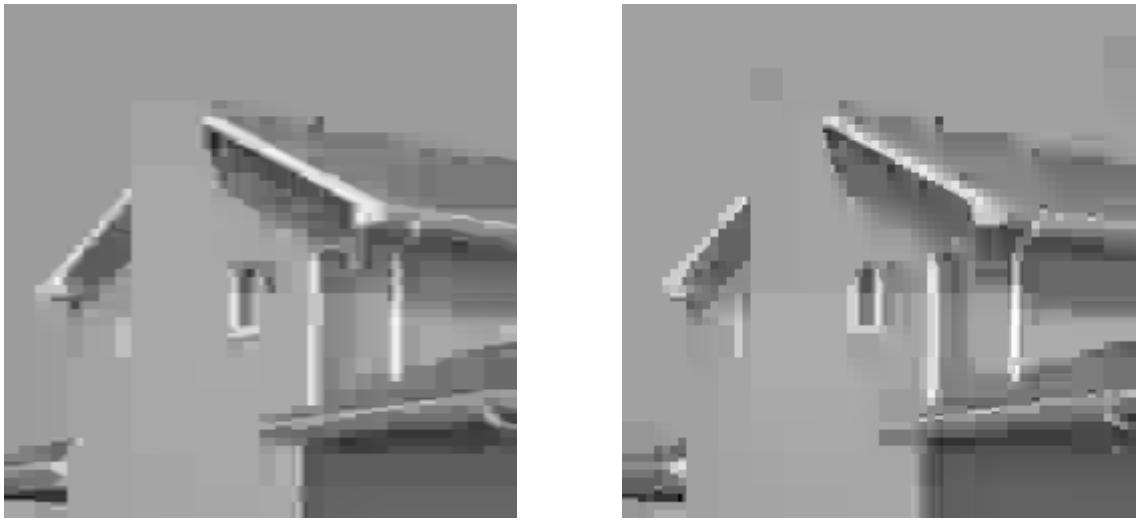


Abb. 8.30: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=45. PSNR dieses Bildes ist 28.23 dB und die Rate 0.04 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=49), PSNR diese Bildes ist 26.896 dB und die Rate 0.041 bpp.

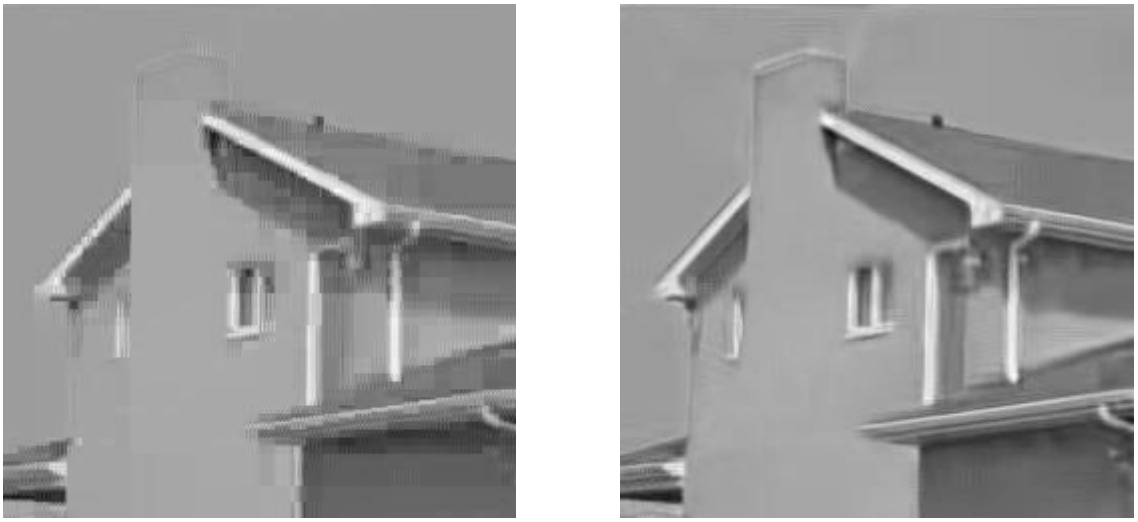


Abb. 8.31: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=45. PSNR dieses Bildes ist 30.24 dB und die Rate 0.71 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=21), PSNR diese Bildes ist 43.624 dB und die Rate 0.73 bpp.



Abb. 8.32: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 31.36 dB und die Rate 0.07 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=44), PSNR diese Bildes ist 30.083 dB und die Rate 0.07 bpp.



Abb. 8.33: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 33.13 dB und die Rate 0.68 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=22), PSNR diese Bildes ist 42.955 dB und die Rate 0.656 bpp.



Abb. 8.34: Das linke Bild ist das Vorhersage-Bild *Haus* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 38.80 dB und die Rate 0.5 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=24), PSNR diese Bildes ist 41.725 dB und die Rate 0.523 bpp.



Abb. 8.35: Das linke Bild ist das Vorhersage-Bild *Haus* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage, Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 42.8 dB und die Rate 1.31 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=16), PSNR diese Bildes ist 47.014 dB und die Rate 1.27 bpp.



Abb. 8.36: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 26.38 dB und die Rate 0.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 26.727 dB und die Rate 0.159 bpp.



Abb. 8.37: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 28.81 dB und die Rate 0.77 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=29), PSNR diese Bildes ist 36.073 dB und die Rate 0.746 bpp.





Abb. 8.38: Das linke Bild ist das Vorhersage-Bild *Lena* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 29.93 dB und die Rate 1 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=26), PSNR diese Bildes ist 38.373 dB und die Rate 0.988 bpp.





Abb. 8.39: Das linke Bild ist das Vorhersage-Bild *Lena* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 35.86 dB und die Rate 1.64 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=20), PSNR diese Bildes ist 42.934 dB und die Rate 1.66 bpp.



Abb. 8.40: Das linke Bild ist das Vorhersage-Bild *Barbara* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 26.55 dB und die Rate 1.15 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=27), PSNR diese Bildes ist 37.714 dB und die Rate 1.183 bpp.



Abb. 8.41: Das linke Bild ist das Vorhersage-Bild *Barbara* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 29.87 dB und die Rate 2.11 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=19), PSNR diese Bildes ist 44.77 dB und die Rate 2.09 bpp.

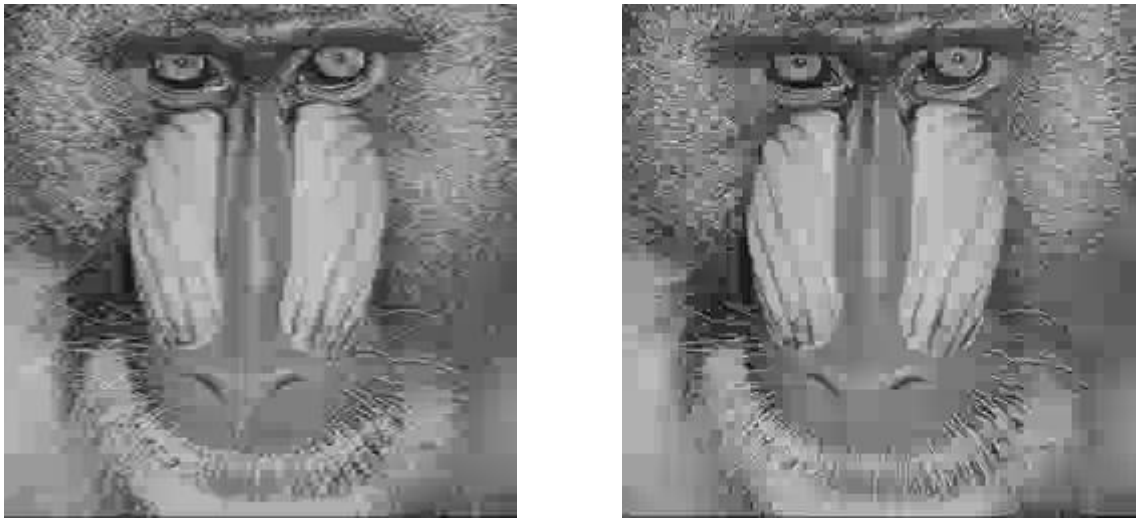


Abb. 8.42: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 23.44 dB und die Rate 0.24 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=43), PSNR diese Bildes ist 23.338 dB und die Rate 0.22 bpp.

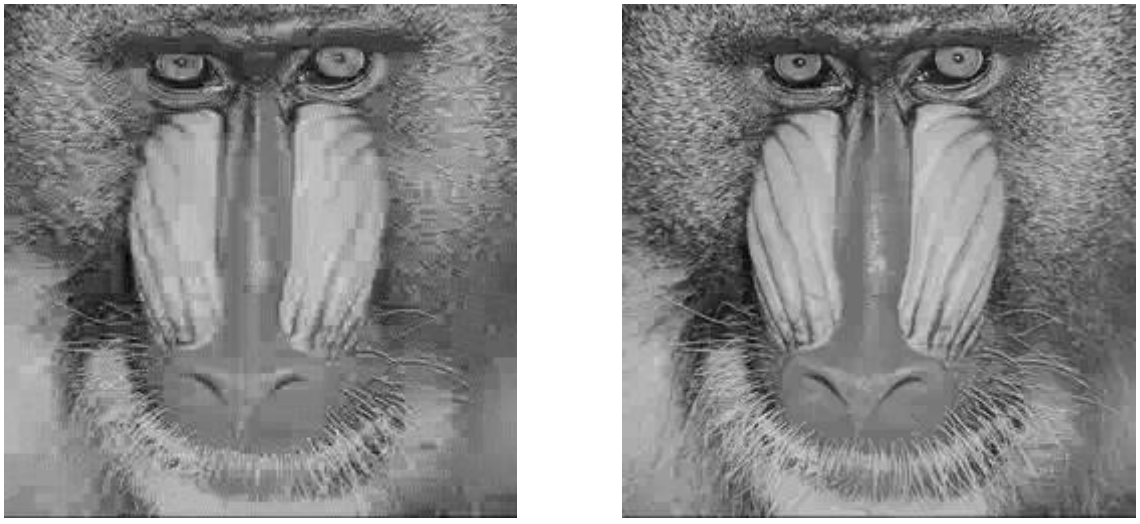


Abb. 8.43: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=40. PSNR dieses Bildes ist 25.28 dB und die Rate 1.02 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=34), PSNR diese Bildes ist 29.52dB und die Rate 1.009 bpp.

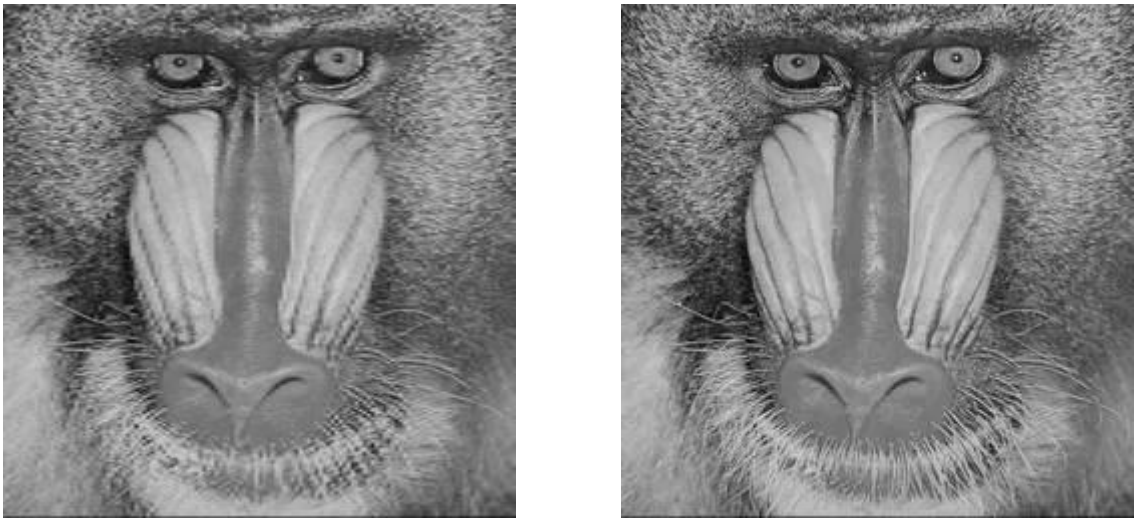


Abb. 8.44: Das linke Bild ist das Vorhersage-Bild *Baboon* bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 27.34 dB und die Rate 1.81 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=28), PSNR diese Bildes ist 34.757 dB und die Rate 1.887 bpp.

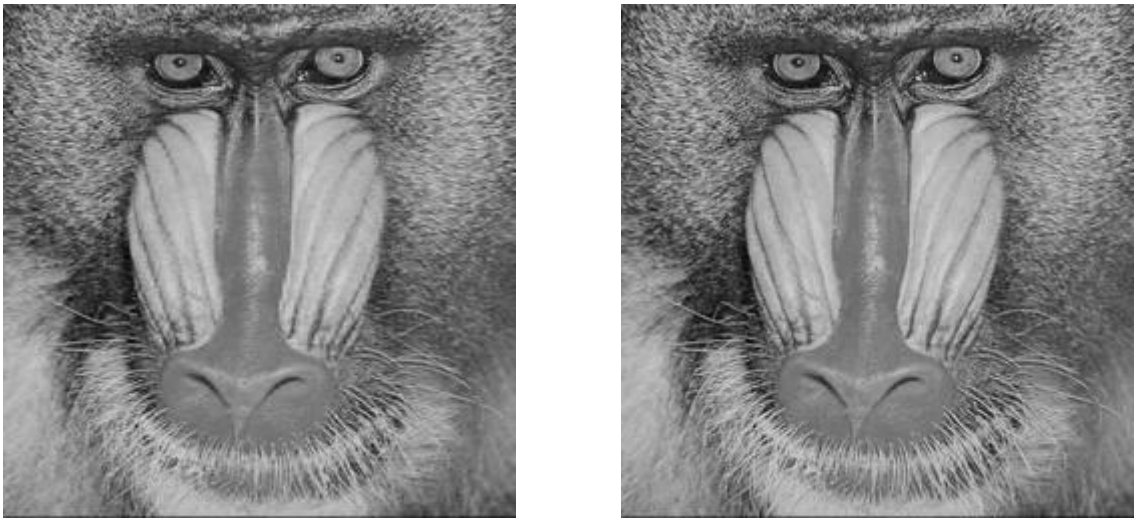


Abb. 8.45: Das linke Bild ist das Vorhersage-Bild *Baboon* nach Korrektur der Bitebenen bei Nutzung verteilter Bildercodierung mit linearer Pixel-Vorhersage. Der Quantisierungsparameter des Schlüssel-Bildes qp=20. PSNR dieses Bildes ist 31.78 dB und die Rate 2.63 bpp. Das rechte Bild wurde komplett mit H.264 codiert (qp=24), PSNR diese Bildes ist 38.47 dB und die Rate 2.555 bpp.

8.4 Visueller Vergleich zwischen Methode I, Methode II, Methode III

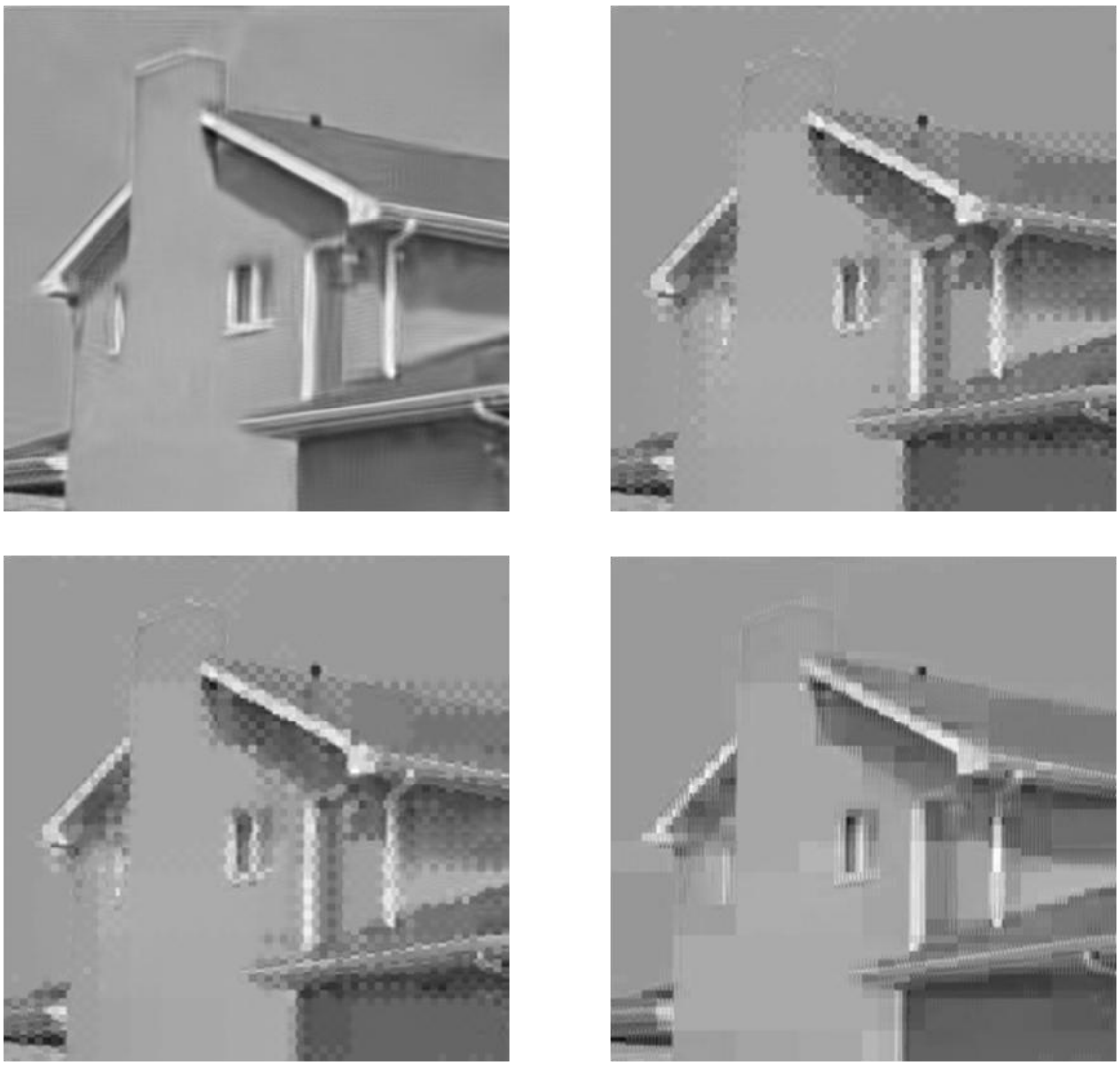


Abb. 8.46: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=50, PSNR=28.05, bpp=0.73), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=50, PSNR=27.93, bpp=0.74), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=50, PSNR=27.47, bpp=0.75).

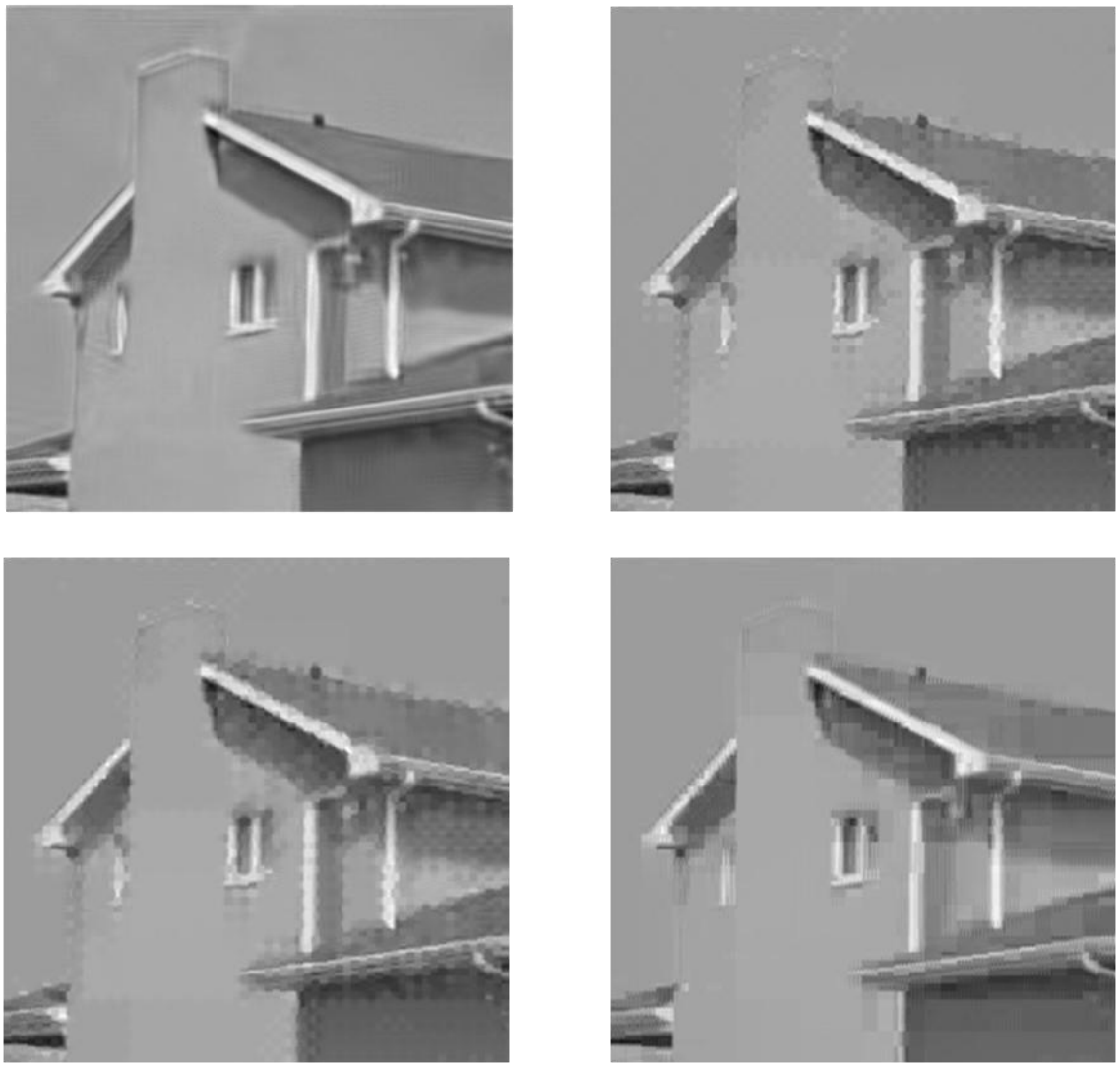


Abb. 8.47: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=31.24, bpp=0.67), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=30.86, bpp=0.74), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=30.24, bpp=0.71).

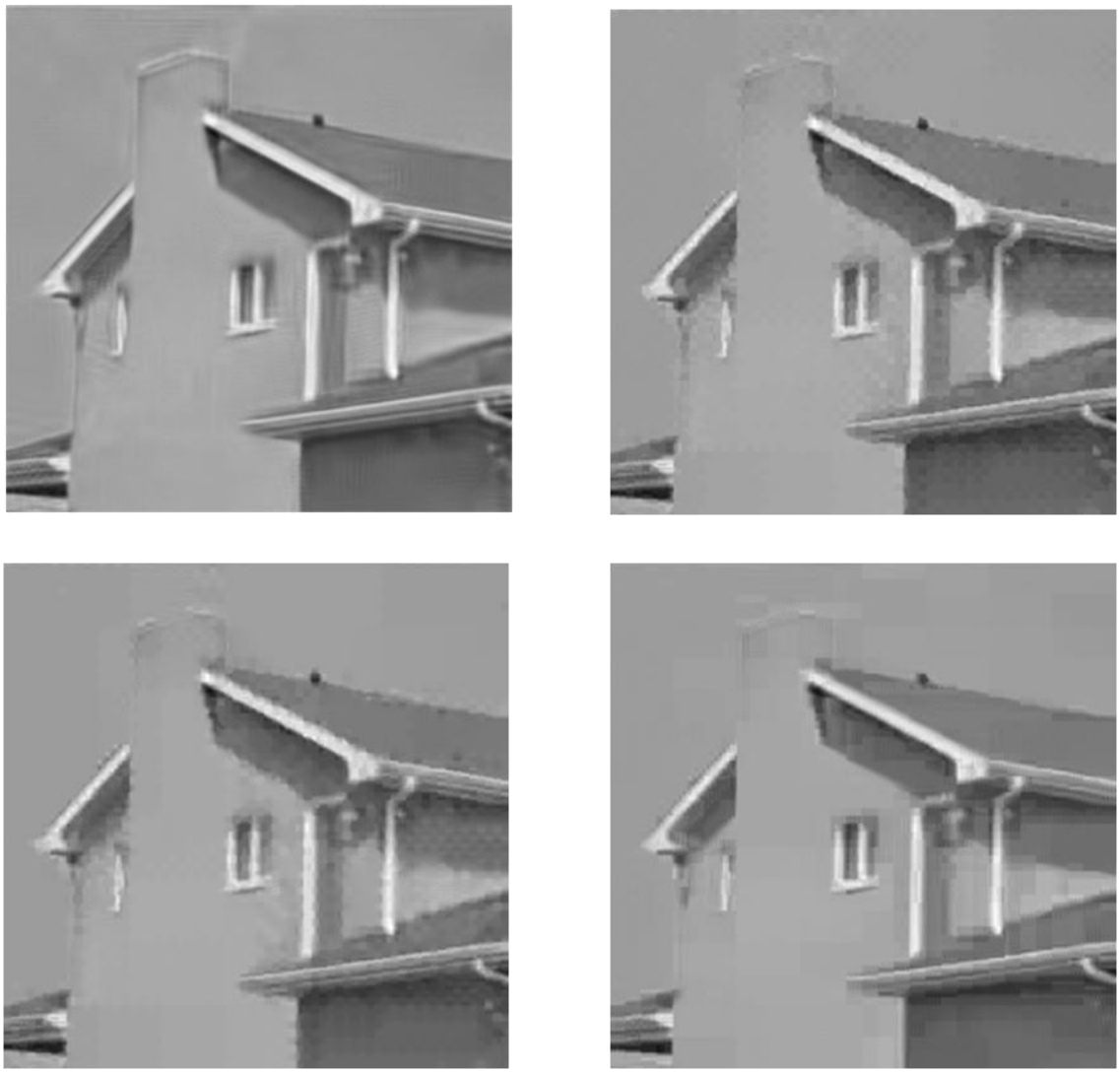


Abb.8.48: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Haus*, Bild *Haus* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=33.87, bpp=0.75), Bild *Haus* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=33.18, bpp=0.75), Bild *Haus* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=33.13, bpp=0.68).



Abb.8.49: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=50, PSNR=24.85, bpp=0.83), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=50, PSNR=23.72, bpp=0.63), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=50, PSNR=23.45, bpp=0.70).



Abb. 8.50: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=27.51, bpp=0.83), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=26.67, bpp=0.71), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=26.13, bpp=0.74).



Abb. 8.51: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Lena*, Bild *Lena* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=35, PSNR=32.63, bpp=0.94), Bild *Lena* nach Korrektur der Bitebenen bei Methode II (qp=35, PSNR=31.18, bpp=0.88), Bild *Lena* nach Korrektur der Bitebenen bei Methode III (qp=35, PSNR=31.42, bpp=0.87).



Abb.8.52: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Barbara*, Bild *Barbara* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=29.04, bpp=1.09), Bild *Barbara* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=26.96,bpp=1.14), Bild *Barbara* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=26.23, bpp=1.06).



Abb.8.53: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Barbara*, Bild *Barbara* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=35, PSNR=31.88, bpp=1.23), Bild *Barbara* nach Korrektur der Bitebenen bei Methode II (qp=35, PSNR=29.03,bpp=1.30), Bild *Barbara* nach Korrektur der Bitebenen bei Methode III (qp=35, PSNR=28.02, bpp=1.20).

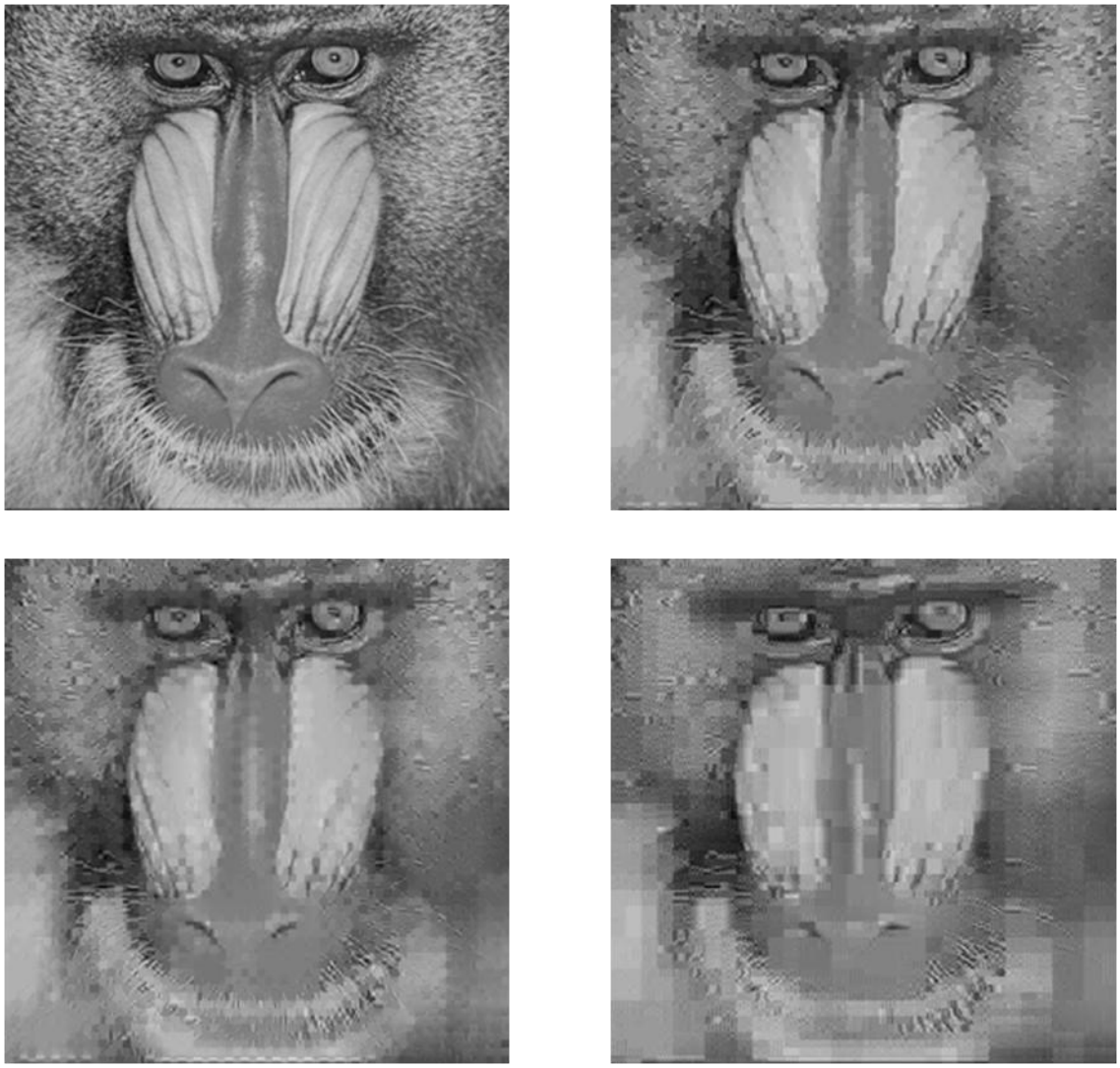


Abb. 8.54: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Baboon*, Bild *Baboon* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=45, PSNR=24.34, bpp=1.12), Bild *Baboon* nach Korrektur der Bitebenen bei Methode II (qp=45, PSNR=23.33,bpp=0.84), Bild *Baboon* nach Korrektur der Bitebenen bei Methode III (qp=45, PSNR=22.88, bpp=0.74).

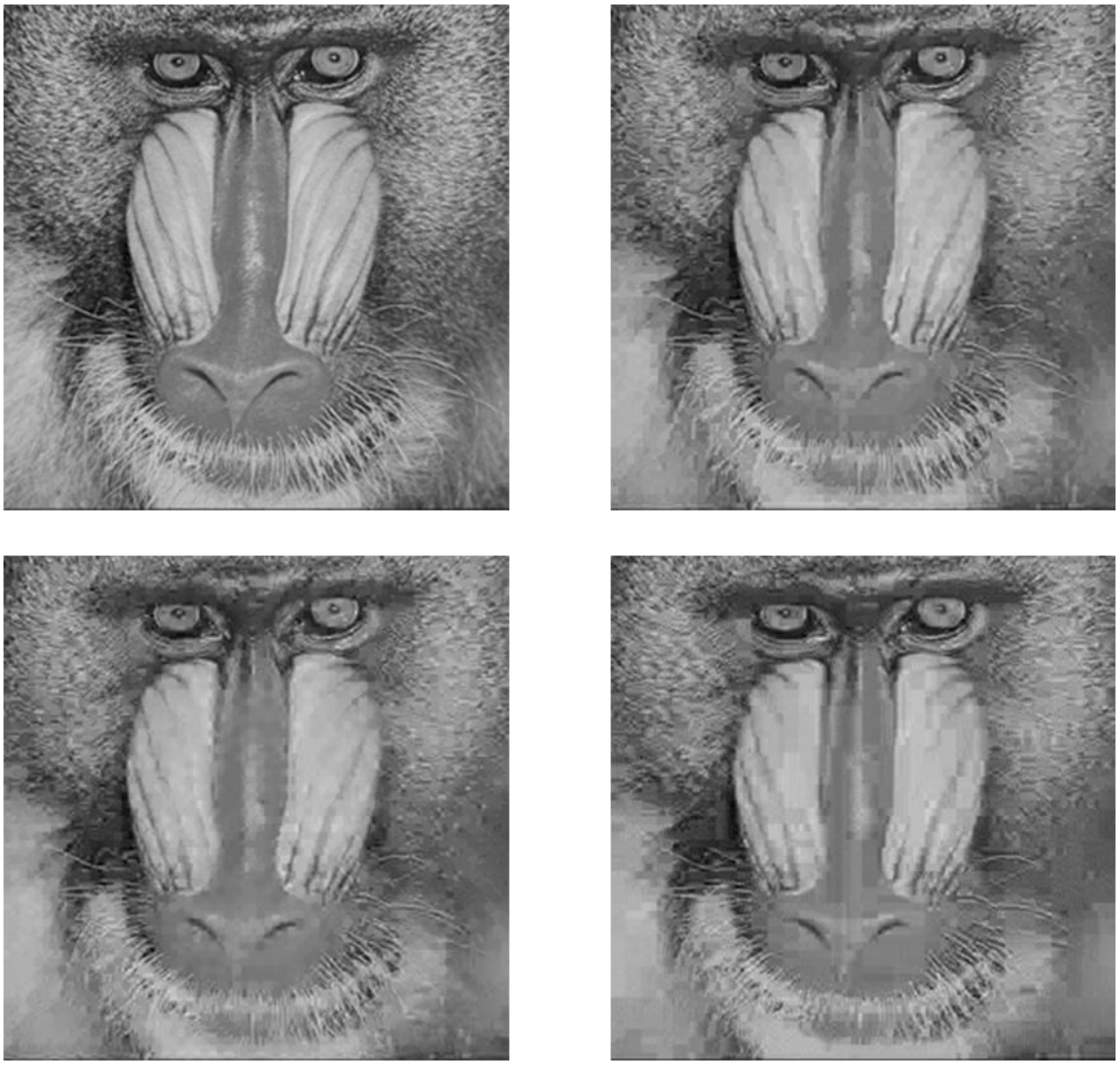


Abb.8.55: Die Bilder von links nach rechts und von oben nach unten sind das original Bild *Baboon*, Bild *Baboon* nach Korrektur der Bitebenen bei Methode I (Der Quantisierungsparameter der Schlüssel-Bilder ist qp=40, PSNR=26.34, bpp=1.22), Bild *Baboon* nach Korrektur der Bitebenen bei Methode II (qp=40, PSNR=25.25,bpp=1.04), Bild *Baboon* nach Korrektur der Bitebenen bei Methode III (qp=40, PSNR=25.28, bpp=1.02).

Bibliographie

[AG02] A. Aaron and B. Girod. Compression with side information using turbo codes. In *Data Compression Conference, 2002. Proceedings. DCC 2002*, pages 252 – 261, 2002.

[AK75] R. Ahlswede and J. Korner. Source coding with side information and a converse for degraded broadcast channels. *Information Theory, IEEE Transactions on*, 21(6):629 – 637, nov 1975.

[Ard04] M. Ardakani. *Effcient Analysis, Design and Decoding of Low-Density Parity-Check Codes*. PhD thesis, University of Toronto, 2004.

[ASG03] A. Aaron, E. Setton, and B. Girod. Towards practical wyner-ziv coding of video. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 3, pages III – 869–72 vol.2, 14-17 2003.

[AZG02] A. Aaron, Rui Zhang, and B. Girod. Wyner-ziv coding of motion video. In *Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on*, volume 1, pages 240 – 244 vol.1, 2002.

[BAP06] Catarina Brites, João Ascenso, and Fernando Pereira. improving transform domain wyner-ziv video coding performance. In *International Conference on Acoustics, Speech, and Signal Processing*, 2006.

[BM01] J. Bajcsy and P. Mitran. Coding for the slepian-wolf problem with turbo codes. In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 2, pages 1400 –1404 vol.2, 2001.

[CDB05] K. Idrissi C. Dikici, R . Guermazi and A. Baskurt. Distributed source coding of still image. In *European Signal Processing Conference*, Antalya, Turkey, September 2005.

[CLME04] T.P. Coleman, A.H. Lee, M. Medard, and M. Effros. On some new approaches to practical slepian-wolf compression inspired by channel coding. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, pages 282 – 291, 23-25 2004.

[Cov75] T. Cover. A proof of the data compression theorem of slepian and wolf for ergodic sources (corresp.). *Information Theory, IEEE Transactions on*, 21(2):226 – 228, mar 1975.

[CVA02] J. Cardinal and G. Van Assche. Joint entropy-constrained multiterminal quantization. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, page 63, 2002.

[CWO08] Ngai-Man Cheung, Huisheng Wang, and A. Ortega. Sampling-based correlation estimation for distributed source coding under rate and complexity constraints. *Image Processing, IEEE Transactions on*, 17(11):2122 –2137, 2008.

[CX04] S. Cheng and Zixiang Xiong. Successive refinement for the wyner-ziv problem and layered code design. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, page 531, mar. 2004.

[DT01] Michael Marcellin David Taubman. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Kluwer International Series in Engineering and Computer Science. Springer Netherlands, har/cdr edition, November 2001.

[FZE04] M. Fleming, Qian Zhao, and M. Effros. Network vector quantization. *Information Theory, IEEE Transactions on*, 50(8):1584 – 1604, aug. 2004.

[Gal62] R. Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21 –28, jan. 1962.

[GD05] N. Gehrig and P.L. Dragotti. Symmetric and asymmetric slepian-wolf codes with systematic and nonsystematic linear codes. *Communications Letters, IEEE*, 9(1):61 – 63, jan. 2005.

[GF01] J. Garcia-Frias. Compression of correlated binary sources using turbo codes. *Communications Letters, IEEE*, 5(10):417 –419, oct 2001.

[GFC04] J. Garcia-Frias and F. Cabarcas. Approaching the slepian-wolf boundary using practical channel codes. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, pages 330 –330, 2004.

[GFC06] Javier Garcia-Frias and Felipe Cabarcas. Approaching the slepian-wolf boundary using practical channel codes. *Signal Process.*, 86(11):3096–3101, 2006.

[GFZ03] J. Garcia-Frias and Wei Zhong. Ldpc codes for compression of multi-terminal sources with hidden markov correlation. *Communications Letters, IEEE*, 7(3):115 – 117, march 2003.

[GLCF04] M. Gonzalez-Lopez, L. Castedo, and J.G. Frias. Bicm for mimo systems using low-density generator matrix (ldgm) codes. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP 04). IEEE International Conference on*, volume 4, pages iv–677 – iv–680 vol.4, may. 2004.

[HM03] Jeongseok Ha and S.W. McLaughlin. Optimal puncturing of irregular low-density parity-check codes. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 5, pages 3110 – 3114 vol.5, 11-15 2003.

[HR09] Yang Han and William E. Ryan. Low-floor decoders for ldpc codes. *Trans. Comm.*, 57(6):1663–1673, 2009.

[JC83] Shu Lin Jr. and Daniel J. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, 1983.

[JDA+07] C. Jones, S. Dolinar, K. Andrews, D. Divsalar, Y. Zhang, and W. Ryan. Functions and architectures for ldpc decoding. In *Information Theory Workshop, 2007. ITW '07. IEEE*, pages 577 –583, sep. 2007.

[JFZ99] A. Jimenez Felstrom and K.S. Zigangirov. Time-varying periodic convolutional codes with low-density parity-check matrix. *Information Theory, IEEE Transactions on*, 45(6):2181 –2191, sep. 1999.

[JN90] Nuggehally S. Jayant and P. Noll. *Digital Coding of Waveforms: Principles and Applications to Speech and Video*. Prentice Hall Professional Technical Reference, 1990.

[Jun55] Dieter Jungnickel. *Codierungstheorie*. Spektrum Akad, 1955.

[JVSV03] C. Jones, E. Valles, M. Smith, and J. Villasenor. Approximate-min constraint node updating for ldpc code decoding. In *Military Communications Conference, 2003. MILCOM 2003. IEEE*, volume 1, pages 157 – 162 Vol.1, oct. 2003.

[KDR01] J. Kusuma, L. Doherty, and K. Ramchandran. Distributed compression for sensor networks. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 82 –85 vol.1, 2001.

[KM79] J. Korner and K. Marton. How to encode the modulo-two sum of binary sources (corresp.). *Information Theory, IEEE Transactions on*, 25(2):219 – 221, mar 1979.

[KNG07] D. Kubasov, J. Nayak, and C. Guillemot. Optimal reconstruction in wyner-ziv video coding with multiple side information. In *Multimedia Signal Processing, 2007. MMSP 2007. IEEE 9th Workshop on*, pages 183 –186, 2007.

[KRR90] P. Yip K. Ramamohan Rao. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press, September 1990.

[KTRR03] P. Koulgi, E. Tuncel, S.L. Regunathan, and K. Rose. On zero-error coding of correlated sources. *Information Theory, IEEE Transactions on*, 49(11):2856 – 2873, nov. 2003.

[Liv02] C.N. Liveris, A.D.; Zixiang Xiong; Georghiades, editor. *A distributed source coding technique for correlated images using turbo-codes*, volume 6 Issue:9. Communications Letters, IEEE, IEEE Communications Society, 2002.

[LLN+04] C.-F. Lan, A.D. Liveris, K. Narayanan, Zixiang Xiong, and C. Georghiades. Slepian-wolf coding of multiple m-ary sources using ldpc codes. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, page 549, 23-25 2004.

[Llo82] S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129 – 137, mar. 1982.

[LMSS01] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman. Improved low-density parity-check codes using irregular graphs. *Information Theory, IEEE Transactions on*, 47(2):585 –598, feb. 2001.

[LS71] H. J. Landau and D. Slepian. Some computer experiments in picture processing for bandwidth reduction. Technical Report 50:5, Bell Sys. Tech. J, 1971.

[LVG09] Yao-Chung Lin, David Varodayan, and Bernd Girod. Distributed source coding authentication of images with contrast and brightness adjustment and affine warping. In *Proceedings of the 27th conference on Picture Coding Symposium*, PCS'09, pages 317–320, Piscataway, NJ, USA, 2009. IEEE Press.

[LXG02] A.D. Liveris, Zixiang Xiong, and C.N. Georghiades. Compression of binary sources with side information at the decoder using ldpc codes. *Communications Letters, IEEE*, 6(10):440 – 442, oct 2002.

[LXG03] A.D. Liveris, Zixiang Xiong, and C.N. Georghiades. Nested convolutional/turbo codes for the binary wyner-ziv problem. In *Proceedings of the International Conference on Image Processing. ICIP 2003.*, volume 1, pages I – 601–4 vol.1, sep. 2003.

[Mac99] D.J.C. MacKay. Good error-correcting codes based on very sparse matrices. *Information Theory, IEEE Transactions on*, 45(2):399 –431, mar. 1999.

[Mac03] David J. C. MacKay, David J. C. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[ME02] D. Muresan and M. Effros. Quantization as histogram segmentation: globally optimal scalar quantizer design in network systems. In *Data Compression Conference, 2002. Proceedings. DCC 2002*, pages 302 – 311, 2002.

[MHKK03] H.S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky. Low-complexity transform and quantization in h.264/avc. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):598 – 603, jul. 2003.

[Moo05] Todd K. Moon. *Error Correction Coding: Mathematical Methods and Algorithms*. John Wiley & Sons, Juli 2005.

[MWD99] D.J.C. MacKay, S.T. Wilson, and M.C. Davey. Comparison of constructions of irregular gallager codes. *Communications, IEEE Transactions on*, 47(10):1449 –1454, oct. 1999.

[Nol99] Peter Noll. *Statistische Nachrichten-Theorie*. Technische Universität Berlin, 1999.

[PKR02] S.S. Pradhan, J. Kusuma, and K. Ramchandran. Distributed compression in a dense microsensor network. *Signal Processing Magazine, IEEE*, 19(2):51 –60, mar. 2002.

[PR99] S.S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (discus): design and construction. In *Data Compression Conference, 1999. Proceedings. DCC '99*, pages 158 –167, 29-31 1999.

[PR00a] S.S. Pradhan and K. Ramchandran. Distributed source coding: symmetric rates and applications to sensor networks. In *Data Compression Conference, 2000. Proceedings. DCC 2000*, pages 363 –372, 2000.

[PR00b] S.S. Pradhan and K. Ramchandran. Geometric proof of rate-distortion function of gaussian sources with side information at the decoder. In *Information Theory, 2000. Proceedings. IEEE International Symposium on*, page 351, 2000.

[PR03] S.S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (discus): design and construction. *Information Theory, IEEE Transactions on*, 49(3):626 – 643, mar 2003.

[QHL07] Lin-Bo Qing, Xiao-Hai He, and Rui Lv. Modeling non-stationary correlation noise statistics for wyner-ziv video coding. In *Wavelet Analysis and Pattern Recognition, 2007. ICWAPR '07. International Conference on*, volume 1, pages 316 – 320, 2007.

[R.G63] R.G.Gallager. *Low density parity check code*. PhD thesis, Cambridge, September 1963.

[Ric10] Iain Richardson. *The H.264 Advanced Video Compression Standard*. John Wiley & Sons, 2 edition, April 2010.

[RLG07] A. Roumy, K. Lajnef, and C. Guillemot. Rate-adaptive turbo-syndrome scheme for slepian-wolf coding. In *Signals, Systems and Computers, 2007. ACSSC 2007. Conference Record of the Forty-First Asilomar Conference on*, pages 545 –549, 4-7 2007.

[RU01a] T.J. Richardson and R.L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *Information Theory, IEEE Transactions on*, 47(2):599 –618, feb. 2001.

[RU01b] T.J. Richardson and R.L. Urbanke. Efficient encoding of low-density parity-check codes. *Information Theory, IEEE Transactions on*, 47(2):638 –656, feb. 2001.

[RU08] T. Richardson and R. Urbanke. *Modern Coding Theory*. Cambridge University Press, Cambridge, March 2008.

[SCV04] S. Sesia, G. Caire, and G. Vivier. Incremental redundancy hybrid arq schemes based on low-density parity-check codes. *Communications, IEEE Transactions on*, 52(8):1311 – 1321, aug. 2004.

[Ser00] S.D. Servetto. Lattice quantization with side information. In *Data Compression Conference, 2000. Proceedings. DCC 2000*, pages 510 –519, 2000.

[SF05] M. Sartipi and F. Fekri. Distributed source coding in wireless sensor networks using ldpc coding: the entire slepian-wolf rate region. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 4, pages 1939 – 1944 Vol. 4, 13-17 2005.

[Sha48] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423,623–656, July, October 1948.

[Sim02] G. Simonyi. On witsenhausen's zero-error rate for multiple sources. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, page 363, 2002.

[SLXG04] V. Stankovic, A.D. Liveris, Zixiang Xiong, and C.N. Georghiades. Design of slepian-wolf codes by channel code partitioning. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, pages 302 – 311, 23-25 2004.

[SLXG06] V. Stankovic, A.D. Liveris, Zixiang Xiong, and C.N. Georghiades. On code design for the slepian-wolf problem and lossless multiterminal networks. *Information Theory, IEEE Transactions on*, 52(4):1495 –1507, april 2006.

[SPR03] D. Schonberg, S.S. Pradhan, and K. Ramchandran. Distributed code constructions for the entire slepian-wolf rate region for arbitrarily correlated sources. In *Signals, Systems and Computers, 2003. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 1, pages 835 – 839 Vol.1, 9-12 2003.

[SW73] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *Information Theory, IEEE Transactions on*, 19(4):471 – 480, jul 1973.

[Tan81] R. Tanner. A recursive approach to low complexity codes. *Information Theory, IEEE Transactions on*, 27(5):533 – 547, sep. 1981.

[TL05] Peiyu Tan and Jing Li. Enhancing the robustness of distributed compression using ideas from channel coding. In *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, volume 4, pages 5 pp. –2389, 2-2 2005.

[TLB05] Zhenyu Tu, Jing Li, and Rick S. Blum. An efficient sf-isf approach for the slepian-wolf source coding problem. In *EURASIP J. ON APPLIED SIGNAL PROCESSING*, pages 961–971, 2005.

[TT91] Joy Thomas.Cover; Thomas. *elements of information theory*. Wiley & Sons, 1991.

[Tun07] E. Tuncel. Kraft inequality and zero-error source coding with decoder side information. *Information Theory, IEEE Transactions on*, 53(12):4810 –4816, dec. 2007.

[WER09] Shu Lin William E. Ryan. *Channel Codes: Classical and Modern*. Cambridge University Press, September 2009.

[WGL05] R.P. Westerlaken, R.K. Gunnewiek, and R.L. Lagendijk. The role of the virtual channel in distributed source coding of video. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 1, pages I – 581–4, 2005.

[WL09] J. Wang and R. Liu. Low complexity dct-based distributed source coding for hyperspectral image. *ChinaCom2009-Information and Coding Theory Symposium*, 1:1 – 5, May 2009.

[Wol73] J.K. Wolf. Data reduction for multiple correlated sources. In *5th Colloquium Microwave Communication*, page 287–295, Jun 1973.

[WSBL03] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h.264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560 –576, jul. 2003.

[WTV08] A.B. Wagner, S. Tavildar, and P. Viswanath. Rate region of the quadratic gaussian two-encoder source-coding problem. *Information Theory, IEEE Transactions on*, 54(5):1938 –1961, may. 2008.

[Wyn74] A. Wyner. Recent results in the shannon theory. *Information Theory, IEEE Transactions on*, 20(1):2 – 10, jan 1974.

[Wyn75] A. Wyner. On source coding with side information at the decoder. *Information Theory, IEEE Transactions on*, 21(3):294 – 300, may 1975.

[WZ76] A. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *Information Theory, IEEE Transactions on*, 22(1):1 – 10, jan 1976.

[WZS93] Y. Wang, Q.-F. Zhu, and L. Shaw. Maximally smooth image recovery in transform coding. *Communications, IEEE Transactions on*, 41(10):1544 –1551, oct 1993.

[YB06] Y.-O. Yan and T. Berger. Zero-error instantaneous coding of correlated sources with length constraints is np-complete. *Information Theory, IEEE Transactions on*, 52(4):1705 –1708, april 2006.

[YDD07] Feng Yang, Qionghai Dai, and Guiguang Ding. Multi-view images coding based on multiterminal source coding. In *ICASSP (1)'07*, pages 1037–1040, 2007.

[Zha07] Y. Zhang. *Design of Low-Floor Quasi-Cyclic IRA Codes and Their FPGA Decoders*. PhD thesis, ECE Dept., University of Arizona, May 2007.

[ZLC08] Jinrong Zhang, Houqiang Li, and Chang Wen Chen. Distributed image coding based on integrated markov random field modeling and ldpc decoding. In *MMSP*, pages 261–266, 2008.

[ZSE02] R. Zamir, S. Shamai, and U. Erez. Nested linear/lattice codes for structured multiterminal binning. *Information Theory, IEEE Transactions on*, 48(6):1250 –1276, jun. 2002.