Vortrag basierend auf dem Paper
Embedded Image Coding Using Zerotrees of Wavelet Coefficients.

Im folgenden wird ein Verfahren zur verlustbehafteten Kompression von Bildern diskutiert. Das Verfahren wurde von Jerome M. Shapiro beschrieben in IEEE Transactions on Signal Processing Vol. 41, No. 12, December 1993. [1]

Weitere Literatur:

Orthogonal pyramid transforms for image coding. E.H. Adelson, E.Simoncelli. Proc. SPIE, Vol. 845, Visual Comm. and Image Processing II, 1987. [2]

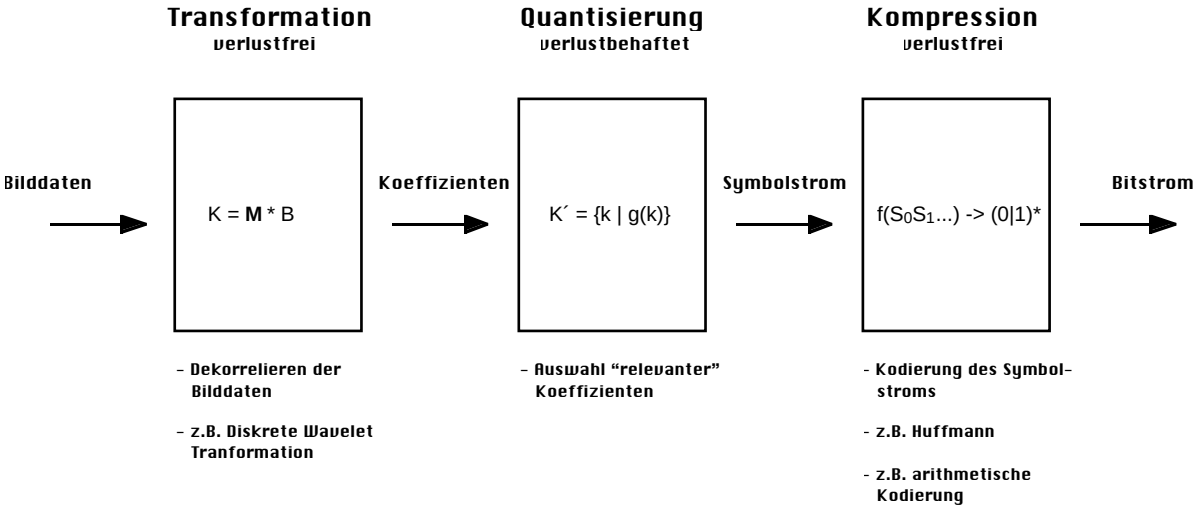
Arithmetic coding for data compression. I.H. Witten, R.M. Neal, J.G. Cleary. Communications of the ACM, June 1987, Volume 30, Number 6. [3]

Inhalt

Verlustbehaftete Bildkompression

Bilder in herkömmlicher Darstellung (RGB, YUV) sind zur **Übertragung auf Kanälen mit niedriger Datenrate** ungeeignet. Es besteht daher ein Bedarf an effizienten Algorithmen zur Kompression von Bilddaten. Hierbei sind verlustbehaftete Verfahren von besonderem Interesse, da die genannten Darstellungen Informationen enthalten, die ein Betrachter nicht wahrnehmen kann und deren Verlust sich somit nicht auf die subjektive Bildqualität auswirkt.

Grundsätzlich lassen sich verlustbehaftete Verfahren in drei Schritte aufteilen:



Das einfachste Beispiel ist die Halbierung der Auflösung. Aus Sicht der Signalverarbeitung entspricht dies einer Tiefpassfilterung. In obiger Darstellung würden bei der Transformation Tiefpasskoeffizienten und Hochpasskoeffizienten getrennt und bei der Quantisierung alle Hochpasskoeffizienten weggeworfen.

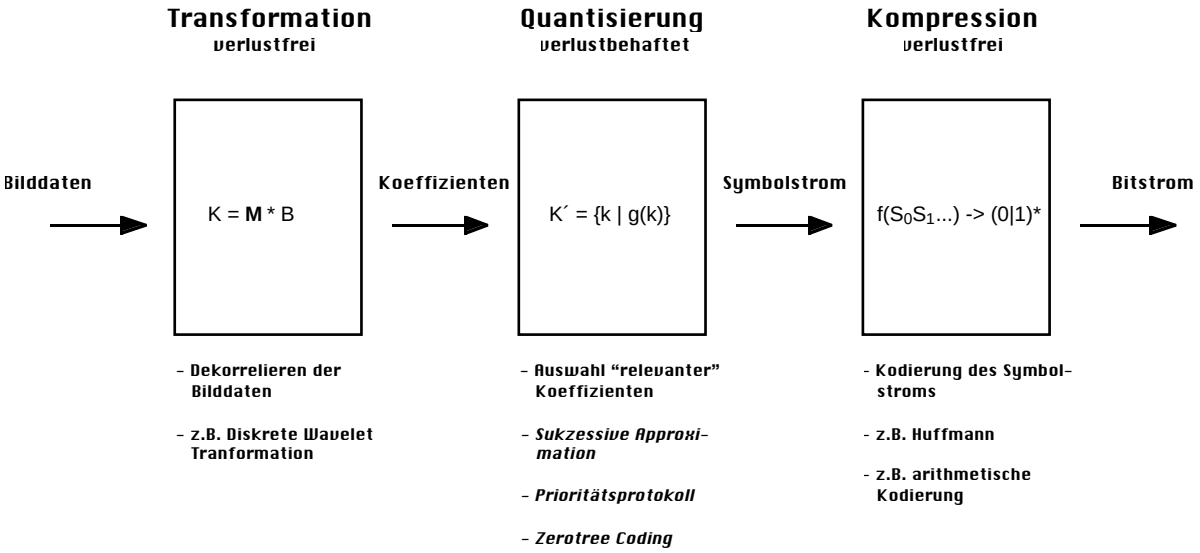
So einfach dieses Beispiel ist, es wiederspiegelt die Grundidee einiger sehr effizienter Verfahren. Die Auflösung des Bildes wird dabei ausgehend von einer niedrigen Auflösung schrittweise verfeinert. Bei dem hier besprochenen Verfahren wird zusätzlich die Erkenntnis genutzt, daß sich anhand eines Bildes niedriger Auflösung das Vorhandensein von Details nur mit mäßigem Erfolg vorhersagen läßt, das Fehlen solcher aber gut gelingt. Vereinfacht ausgedrückt: In einem sehr hellen oder sehr dunklen Bereich gibt es vermutlich keine sichtbaren Details. Was für diese beiden Extrembeispiele gilt, gilt oft auch für Flächen ohne große Änderungen.

Eingebettete Verfahren

Bei Kanälen mit sehr niedrigen Bitraten ist es wünschenswert eine **Vorabversion des Bildes** bereits **während der Übertragung** darstellen zu können, die dem Schlußresultat so nahe wie möglich kommt und sich im Idealfall **mit jedem empfangenen Bit verbessert**. Dabei sollten keine Informationen übertragen werden die später nicht mehr gebraucht werden. Von einem anderen Blickwinkel betrachtet wird jeweils die Differenz zwischen dem bereits übertragenen Bild (anfänglich uni-grau) und dem tatsächlichen Bild kodiert. Der Empfänger kann die Übertragung abbrechen sowie ihm die Qualität genügt.

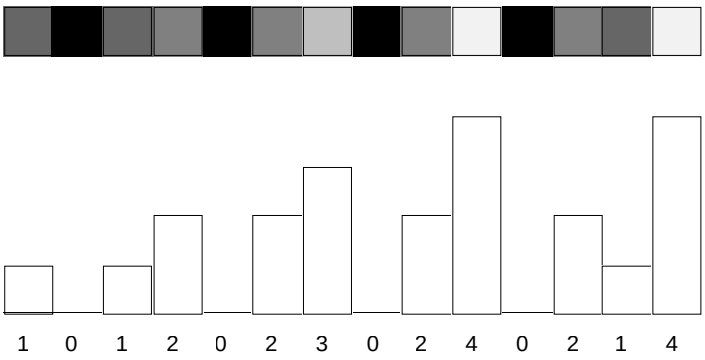
Leider liefern Verfahren die strikt die Auflösung verfeinern keine befriedigenden Resultate. Dies liegt vor allem daran, daß wir die Existenz einer Kante oder scharfen Linie stärker wahrnehmen als den Helligkeitsunterschied zweier Flächen. In niedrigen Auflösungen werden jedoch solche Details oder Anomalien nicht wiedergegeben, vielmehr werden nur statistische Trends angezeigt.

Ein gutes sollte somit die aus der Transformation erhaltenen Koeffizienten mit einem Prioritätsprotokoll entsprechend ihrer Signifikanz für den optischen Eindruck ordnen und übertragen.



Transformation

Das Ziel der Transformation ist die Überführung der stark korrelierten Bilddaten in unkorrelierte Koeffizienten, welche dann im Quantisierungsschritt unabhängig voneinander betrachtet werden können. Um dies zu verdeutlichen betrachten wir zunächst eine einzelne Bildzeile.



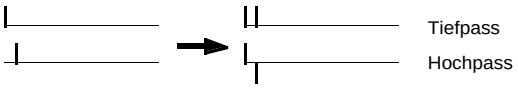
Man kann sich diese Bildzeile vorstellen als gewichtete Summe von n Einheitsvektoren:

```
  1 x ( 1  0  0  0  0  0  0  0  0  0  0  0  0  0 )
  0 x ( 0  1  0  0  0  0  0  0  0  0  0  0  0  0 )
  1 x ( 0  0  1  0  0  0  0  0  0  0  0  0  0  0 )
  2 x ( 0  0  0  1  0  0  0  0  0  0  0  0  0  0 )
  0 x ( 0  0  0  0  1  0  0  0  0  0  0  0  0  0 )
  2 x ( 0  0  0  0  0  1  0  0  0  0  0  0  0  0 )
  3 x ( 0  0  0  0  0  0  1  0  0  0  0  0  0  0 )
  0 x ( 0  0  0  0  0  0  0  1  0  0  0  0  0  0 )
  2 x ( 0  0  0  0  0  0  0  0  1  0  0  0  0  0 )
  4 x ( 0  0  0  0  0  0  0  0  0  1  0  0  0  0 )
  0 x ( 0  0  0  0  0  0  0  0  0  0  1  0  0  0 )
  2 x ( 0  0  0  0  0  0  0  0  0  0  0  1  0  0 )
  1 x ( 0  0  0  0  0  0  0  0  0  0  0  0  1  0 )
+ 4 x ( 0  0  0  0  0  0  0  0  0  0  0  0  0  1 )
-------------------------------------------------
    1  0  1  2  0  2  3  0  2  4  0  2  1  4
```

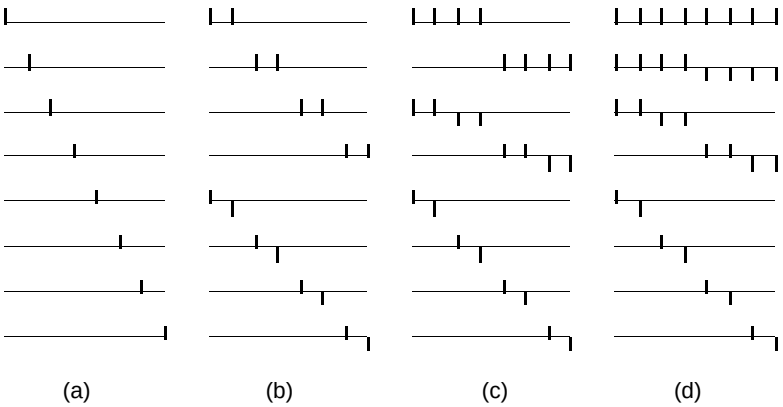
Anstelle dieser einfachen Einheitsvektoren kann man beliebige n orthogonale Basisvektoren verwenden und die Zeile als Linearkombination dieser Vektoren darstellen. Diese Vektoren können zum Beispiel Wellencharakter haben wodurch sich weniger scharfe Übergänge ergeben.

Haar-Transformation

Bei der Haar-Transformation faßt man beispielsweise je zwei Einheitsvektoren zusammen und trennt sie in einen Hochpass- und einen Tiefpassteil.



Mit je zwei Tiefpassvektoren wiederholt man diesen Schritt nun fortlaufend wiederholt, während man die Hochpassvektoren in das Basisset übernimmt.



(a) Ausgangsset

(d) endgültiges Basisset

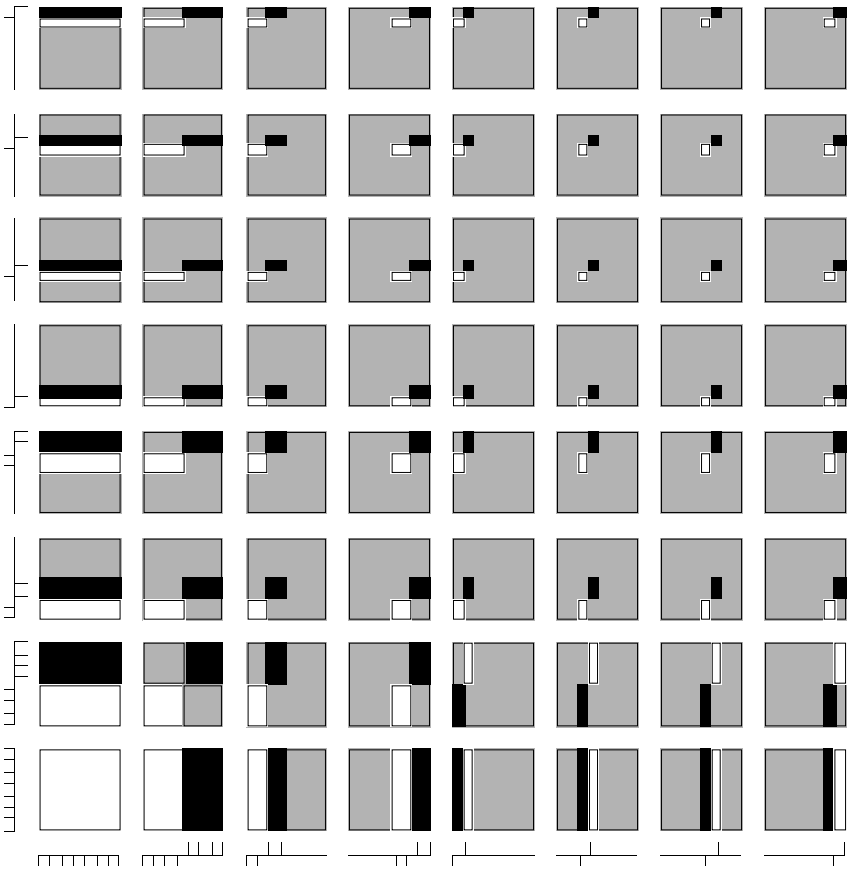
Betrachtet man das entstandene Basisset, so stellt man fest, daß es die Bildzeile in verschiedene Auflösungen zerlegt:

- Der Koeffizient des ersten Vektors gibt den Mittelwert der Helligkeit aller Bildpunkte an und somit die Auflösung 1.
- Der zweite Koeffizient beschreibt den Unterschied zwischen der linken und der rechten Bildhälfte und ergibt zusammen mit dem ersten Koeffizienten die Auflösung 2.
- Der 3. und 4. Koeffizient geben je die Unterschiede einer Hälfte an und ergeben mit dem 1. und dem 2. die Auflösung 4.
- usw.

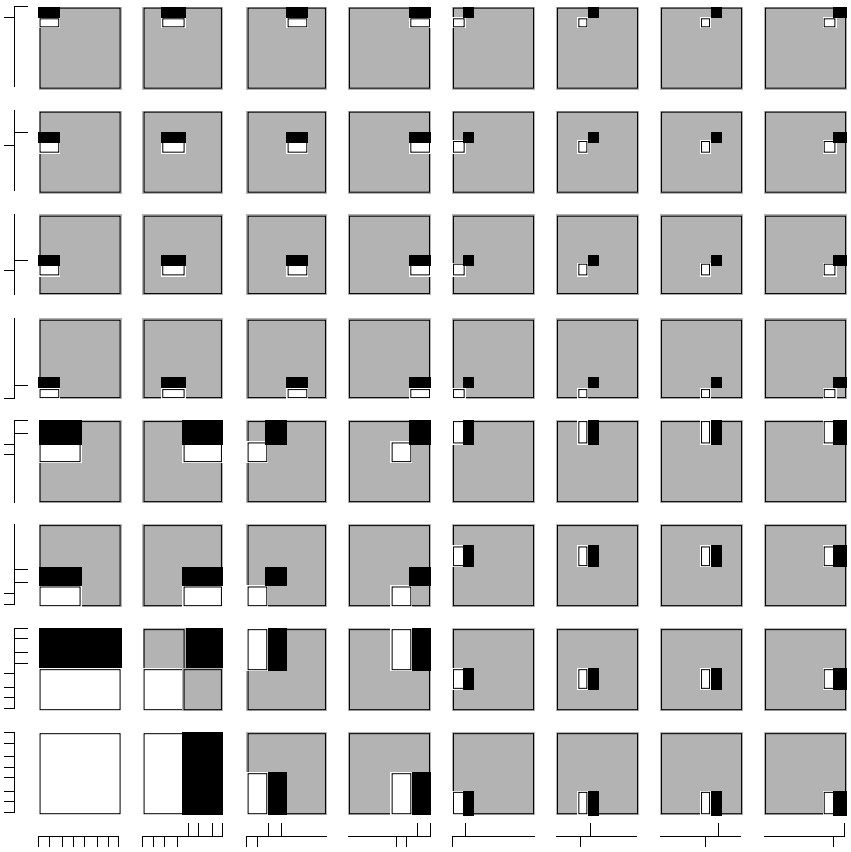
Leider ist die Haar-Transformation für Bilddaten nicht sehr geeignet; die Vorgehensweise je zwei Tiefpassvektoren zusammenzufassen und daraus einen neuen Tiefpass- und einen Hochpassvektor zu erzeugen ist bei allen Verfahren gleich. Ein gutes **Basisset** sollte **selbstähnlich** sein und eine **hohe Lokalität im Raum- und im Frequenzbereich** aufweisen.

2-D Fall

Bilddaten sind in X- und in Y-Richtung korreliert. Es ist daher sinnvoll ein Bild mit n x m Pixeln als Summe von eben solchen Matrizen anzuschauen. Kombiniert man die Muster des Basisset der Haar-Transformation in X-Y-Richtung so erhält man die folgenden Matrizen (schwarz +1, weiß -1, grau 0):



Dieses zweidimensionale Basisset verletzt die Forderung nach der Selbstähnlichkeit. Generell zeigt sich, daß es sinnvoller ist, direkt die 2D-Lowpass-Komponenntten zu zerlegen. Dadurch ergeben sich nur 3 verschiedene Abtastmuster, die sich in verschiedenen Größen und an verschiedenen Stellen wiederholen (Auflösungspyramide) und für je eine der drei Abtastrichtungen horizontal, vertikal oder diagonal stehen.



Jede solche Matrix ist nun eben gerade eines jener berühmt berüchtigten Wavelets (Wellchen).

Um die Umrechnung der Bilddaten in Koeffizienten und zurück zu verstehen, schreiben wir die Matrizen der Bilddaten und des Basisset als Vektoren:

Bilddaten (2 x2 Pixel)

$$\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} \longrightarrow \begin{bmatrix} b_{00} & b_{01} & b_{10} & b_{11} \end{bmatrix}$$

Matrizen-Basisset

$$M_{10} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \qquad M_{11} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

$$M_{00} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \qquad M_{01} \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

$$M_{00} \longrightarrow m_{00} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

$$M_{10} \longrightarrow m_{10} = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}$$

$$M_{01} \longrightarrow m_{01} = \begin{bmatrix} -1 & 1 & -1 & 1 \end{bmatrix}$$

$$M_{11} \longrightarrow m_{11} = \begin{bmatrix} 0 & 1 & -1 & 0 \end{bmatrix}$$

Aus den Koeffizienten K_{00} .. K_{nn} erhält man die Bilddaten durch Multiplikation mit der Matrix M welche als Kolonnen die Vektoren m_{ij} hat.

$$\mathbf{M} = \begin{bmatrix} \downarrow m_{00} & \downarrow m_{01} & \downarrow m_{10} & \downarrow m_{11} \\ & & & \end{bmatrix}$$

$$\mathbf{M} * \vec{k} = \vec{b} \quad \Rightarrow \quad \vec{k} = \mathbf{M}^{-1} * \vec{b}$$

Woraus sich ergibt, daß die Koeffizienten durch Multiplikation des Bilddaten-vektors mit der inversen Matrix M^{-1} berechnet werden können. Dies setzt jedoch voraus, daß M invertierbar ist, was genau dann der Fall ist, wenn das Basisset orthogonal ist.

Für eine große numerische Stabilität ist es außerdem wünschenswert, daß das Basisset aus Vektoren $\vec{x}$ besteht mit $\sum_i x_i = 1$, da hierdurch die numerische Auflösung der Koeffizienten ideal genutzt wird.

Geht man von einem Bild mit 1000 x 1000 Pixeln aus, so besitzt die oben erwähnte Matrix M 1000^4 Koeffizienten und für die Transformation müßten im allgemeinen Fall 1000^4 Multiplikationen ausgeführt werden, was natürlich völlig unrealistisch ist. Dies verstärkt die Forderung daß das Basisset eine hohe Lokalität im Raumbereich (die meisten Koeffizienten von M sind Null) und im Frequenzbereich (die meisten Koeffizienten von M^{-1} sind Null) aufweisen muß.

Einige Transformationsverfahren

Diskrete Kosinus Transformation (DCT) in diskreten Blöcken (harter Block)
- Artefakte an den Blockgrenzen
- Harte Blockgrenzen ergeben schlechte Frequenzlokalität bezogen auf das Gesamtbild

Gabor (1946): Komplexer Sinoid * Gaussfenster (weicher Block)
- Basisset nicht orthogonal

Burt und Adelson (1983): Gaussche Pyramide
+ Selbstähnlichkeit
- Basisset nicht orthogonal | Koeffizienten | = 4/3 * | Pixel |

QMF-Pyramide
+ Basisset orthogonal | Koeffizienten | = 4/3 * | Pixel |
+ orthonormal
+ gute Frequenzlokalität
+ Selbstähnlichkeit führt zu einer hierarchischen Zerlegung

Quadrature-Mirror-Filter

Ein sehr gutes Basisset erhält man bei Verwendung von Quadratur-Mirror-Filtern mit einer ungeraden Anzahl taps (Koeffizienten ≠ Null) als Basisfunktionen. Die ungerade Anzahl taps ergibt symmetrische Hochpassfilter was zu einer guten Behandlung der Ränder beiträgt.

Bei den Tiefpassvektoren des Startset werden die taps um je zwei Pixel versetzt. Die Hochpassvektoren werden um je einen Pixel gegenüber den Tiefpassvektoren versetzt. Wobei für die taps t_i, h_i der Tief- bzw. Hochpassvektoren gilt:

$$h_i = -1^n * t_i \,,\ \sum_i h_i = 0.$$

Die zweite Eigenschaft ist bei der schrittweisen Rekonstruktion nützlich, da hierbei schrittweise Hochpassvektoren verschiedener Auflösung auf das Bild addiert werden. Die angegebene Eigenschaft führt gerade dazu, daß die Gesamthelligkeit des Bildes konstant bleibt.

Quantisierung

Im Quantisierungsschritt findet ein **kontrollierter Informationsverlust** statt. Die aus der Transformation gewonnenen Koeffizienten werden dabei bewertet und die Signifikanten werden, gegebenenfalls mit verringerter Präzision, dem Kodierer zugeführt, während die nichtsignifikanten zu Null gesetzt werden.

Bei herkömmlichen Verfahren basiert die Bewertung meist auf Erfahrungswerten. Um diese zu erhalten wird eine große Anzahl Bilder transformiert und jene Koeffizienten ermittelt welche häufig bedeutend sind. Die so ermittelten Signifikanzen bewerten jedoch die Trends eines Bildes zu stark, da Anomalien wie scharfe Linien bei statistischer Betrachtung als Rauschen auftreten.

Beim hier besprochenen Verfahren werden die Koeffizienten einzeln und deterministisch bewertet. Durch diese Strategie werden Fine-Scale-Koeffizienten besser erhalten. Durch geeignete Skalierung des Basisset erreicht man eine nahezu unitäre Transformationsmatrix, welche die L_2-Norm erhält. Es macht daher Sinn alle Koeffizienten mit dem selben Schwellwert zu vergleichen, was die Implementierung vereinfacht und, wie wir noch sehen werden, bei der schrittweisen Annäherung sehr nützlich ist.

Die Kehrseite der Medaille ist, daß nun eine immense Anzahl potentiell signifikanter Koeffizienten, von denen die meisten Null sind, kodiert werden muß. Bei den angestrebten Datenraten von unter einem Bit pro Pixel müssen mehrere Nullkoeffizienten mit einem einzigen Bit kodiert werden.

Significance-Map

Die Gesamtkosten für die Kodierung ergeben sich als Summe der Kosten für die Nullkoeffizienten und der Kosten für die Koeffizienten ungleich Null oder anders betrachtet einer Karte welche Koeffizienten signifikant sind und deren Werte.

$$K_{Total} = K_{Significance\ Map} + K_{Nonzero\ Values}$$

Im folgenden nehmen wir an, daß die Koeffizienten mittelwertfreie symmetrische Zufallsvariablen sind, was für die beschriebene Wavelet-Transformation in guter Näherung zutrifft.

Koeffizienten ungleich Null sind mit gleicher Wahrscheinlichkeit positiv oder negativ, es macht daher Sinn das Vorzeichen der signifikanten Koeffizienten in der Significance-Map SM zu kodieren. Einige Beispiele sollen verdeutlichen wie groß der Kostenanteil der SM bei herkömmlicher Kodierung ist. Sei p die Wahrscheinlichkeit für die Nichtsignifikanz eines Koeffizienten so ist die Entropierate

$$H = -p \log_2 p - (1-p) \log_2 (1-p) + (1+p)(1 + H_{NZ}).$$

Also die Summe der binären Entropie für die Signifikanz eines Koeffizienten plus der Entropie der signifikanten Koeffizienten selbst. Wobei H_{NZ} die Entropie des Absolutbetrags der signifikanten Koeffizienten ist.

Beispiel 1

Gegeben ein 3-Stufen-Quantisierer (-1, 0, 1) und eine Entropierate von 0.5 Bit pro Pixel.

$$pmin(H_{NZ} = 0, H = 0.5) = 0.916$$

1) 91.6 % der Koeffizienten müssen Null sein
2) 83 % des Bitbudgets werden für die Significance Map verwendet

Beispiel 2

Gegeben ein 33-Stufen-Quantisierer (-16, .. , 0, .. , 16) bei gleicher Entropierate.

$$pmin(HNZ = 4, H = 0.5) = 0.954$$

1) 95.4 % der Koeffizienten müssen Null sein
2) 54 % des Bitbudgets werden für die Significance Map verwendet

Je kleiner die Entropierate, also die Anzahl zur Verfügung stehender Bits pro Pixel, um so größer werden die Kosten für die SM.

Zerotree Darstellung

Zur optimalen Kodierung der Significance-Map verwenden wir die Tatsache, daß die verwendete QMF-Transformation selbstähnliche Wavelets benutzt und somit eine hierarchische Zerlegung in mehrere Auflösungen generiert. Es gelingt daher mit guter Trefferquote nichtsignifikante Koeffizienten in feinen Auflösungen vorherzusagen.

Aufgrund der Selbstähnlichkeit der Zerlegung ist es vernünftig anzunehmen, daß der **Betrag der Koeffizienten der selben Richtung** in einem festen Bildbereich **mit feineren Auflösungen abnimmt**, falls nicht so ist dies eine wichtige Information (beispielsweise eine Linie oder Kante) deren Kodierung "aufwendig" sein darf.

Mathematischer Hintergrund: Man betrachtet die Koeffizienten der selben Richtung und zunehmender Auflösung als Zufallsvariablen mit gleicher Dichteverteilung aber abnehmendem Mittelwert. Ist die Varianz eines bezüglich Schwellwert T nicht signifikanten Koeffizienten kleiner T, so ist die beste lineare Schätzung für alle Koeffizienten feinerer Auflösungen, daß diese ebenfalls nicht signifikant sind. Was einleuchtet und hier nicht vorgerechnet werden soll (siehe [1]).

Für die Darstellung der SM definiert das EZW-Verfahren daher ein spezielles Symbol zerotree-root welches besagt, daß der aktuelle Koeffizient und alle Koeffizienten feinerer Auflösungen und selber Richtung (Söhne) Null beziehungsweise nicht signifikant sind. Zusätzlich wird ein Symbol isolated-zero nötig welches einen Nullkoeffizienten kodiert, der mindestens einen Sohn ungleich Null hat. Algorithmich betrachtet ist dies eine Divide-and-Conquer-Strategie zum Suchen nichtsignifikanter Flächen. Im Gegensatz zu JPEG gibt es keine fixen Blockgrößen was zu deutlichen Codierungsgewinnen führt.

Zusammenfassend ergeben sich für die Kodierung der Significance-Map vier Symbole für grobe Auflösungen (pos, neg, zero, zerotree-root) und drei Symbole für die feinste Auflösung (pos, neg, zero).

Schrittweise Annäherung

Um mit den vorgestellten Methoden einen eingebetteten Algorithmus zu entwickeln ist es nötig ein Protokoll zu definieren daß die Koeffizienten nach absteigender Signifikanz überträgt. Hierzu bieten sich grundsätzlich zwei Varianten an.

1) Übertragen der Koeffizienten in absteigender Signifikanz mit jeweils voller Präzision.

2) Abwechselnde Übertragung der Most-Significant-Bits der signifikanten Koeffizienten.

Das EZW-Verfahren verwendet die zweite Variante, da hierbei die **Unsicherheitsintervalle aller Koeffizienten für den Dekodierer gleich groß** sind. Die erste Variante würde zudem die Frage aufwerfen ob beispielsweise das Least-Significant-Bit des 'wichtigsten' Koeffizienten mehr Information trägt als das Most-Significant-Bit des 'zweit wichtigsten' Koeffizienten.

Das Verfahren läuft in drei sich abwechselnd wiederholenden Schritten ab. Im **dominant-pass werden** alle **Koeffizienten** deren Betrag größer als ein Schwellwert T_i ist **als signifikant erkannt**, eine entsprechende SM erstellt und in Zerotree-Darstellung kodiert. Die Beträge der signifikanten Koeffizienten werden in einer subordinat-list eingetragen und in der Matrix der Koeffizienten zu Null gesetzt. Dies damit sie bei einem späteren Durchgang mit niedrigerem Schwellwert $T_j < T_i$ nicht die Kodierung eines zerotree-root verhindern.

Durch die Wahl von $T_{i+1} = T_i / 2$ halbiert sich das Unsicherheitsintervall über alle bisher nicht als signifikant erfaßten Koeffizienten in jedem Durchgang.

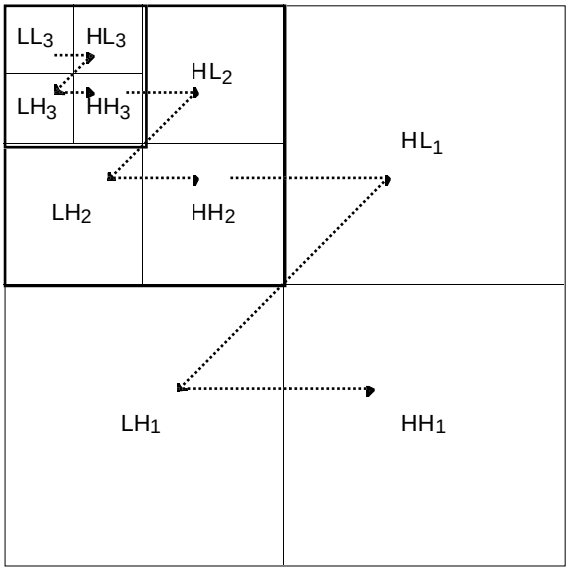
Im anschließenden **subordinate-pass** wird das **i-te Bit aller Koeffizienten** in der subordinate-list übertragen (most-significant-bit zuerst). Hierdurch halbiert sich für den Dekodierer die Unsicherheit über alle Koeffizienten auf der subordinate-list. Durch entsprechende Wahl von T_0 ist das Unsicherheitsintervall über die bisher nicht signifikanten Koeffizienten und die als signifikant erkannten Koeffizienten stets gleich groß (und zwar = T_i).

Nach dem subordinate-pass wird noch eine **Sortierung der subordinate-list** durchgeführt um die Unsicherheit über große (signifikantere) Koeffizienten im nächsten subordinate-pass zuerst zu verringern. Dabei dürfen die Koeffizienten jedoch nur gemäß den dem Dekodierer bereits bekannten Bits sortiert werden.

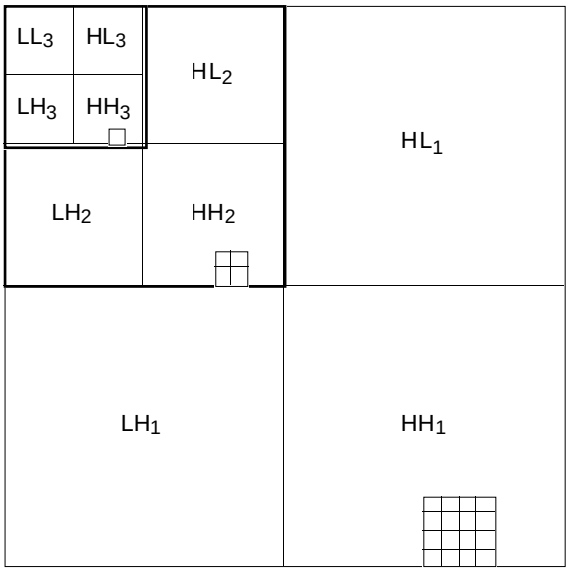
Das EZW-Verfahren verwendet für die Rekonstruktion der Koeffizienten auf der Dekodiererseite die Mitte des Unischerheitsintervalls. Dies könnte gegebenenfalls verbessert werden mit Hilfe von Schätzern für die Koeffizientenverteilung.

Prioritätsprotokoll

Die Reihenfolge in der die Koeffizienten im dominant-pass auf Signifikanz getestet werden ist kritisch. Aus den Überlegungen zur Zerotree-Darstellung der Significance-Map ist klar, daß die Signifikanz der Koeffizienten einer groben Auflösung vor denen der selben Richtung und feineren Auflösung kodiert werden muß. Die folgende Abbildung zeigt die von Shapiro beschriebene Kodierreihenfolge.

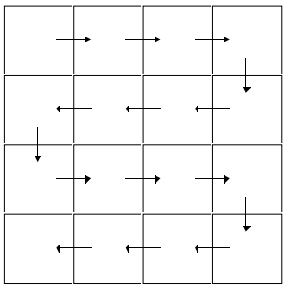


Die Pfeile zeigen die Reihenfolge an, in der noch nicht (mittels zerotree-root) kodierte Signifikanzen kodiert werden. Wobei die Koeffizienten LL_i, HL_i, LH_i zur Auflösung i gehören. Oben links stehen somit in diesem Diagramm die Koeffizienten der gröbsten Auflösung. In jeder feineren Auflösungsstufe vervierfacht sich die Anzahl der Koeffizienten.



Die Abtastreihenfolge der Koeffizienten der jeweils gleichen Reihenfolge ist im Paper nicht beschrieben. Es ist nicht klar ob ebenfalls eine Z-Abtastung statt-

findet. Da bei der späteren Kodierung nur der Vater (gröbere Auflösung) und der Vorgänger (selbe Auflösung) betrachtet werden wäre eine links rechts Strategie vermutlich effizienter.



Faßt man dominant- und subordinate-pass zusammen, so ergeben sich für die Reihenfolge der Verfeinerung von Koeffizienten folgende Prioritäten:

1) Präzision:
   Alle Unsicherheitsintervalle werden halbiert bevor eines erneut betrachtet wird.

2) Betrag des Koeffizienten:
   D) Größere Werte werden früher als signifikant erkannt.
   S) Größere Werte werden zuerst verfeinert.

3) Auflösung:
   Bei der Suche nach signifikanten Koeffizienten werden zunächst alle Koeffizienten einer Auflösung betrachtet, bevor der Algorithmus zu denen der nächst feineren schreitet.

4) Räumliche Anordnung:
   D) Abtastreihenfolge bei der Suche nach signifikanten Koeffizienten.
   S) Koeffizienten mit gleichem Unsicherheitsintervall ändern ihre relative Anordnung nicht.

Kodierung

Das EZW-Verfahren verwendet sehr kleine Symbolalphabete und benötigt eine Kodierung von im Mittel weniger als einem Bit pro Symbol. Dies leistet die weitverbreitete Huffman-Kodierung nur über Umwege, wie das Zusammenfassen von Gruppen von Symbolen.

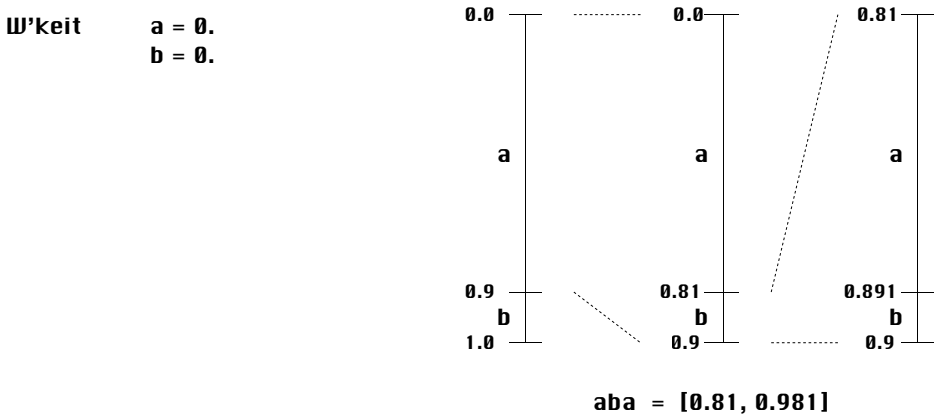
Arithmetische Kodierung

Bei der arithmetischen Kodierung wird eine Nachricht als Teilintervall von [0,1] in den rationalen Zahlen dargestellt. Dazu werden die möglichen Symbole von 1 bis n numeriert und die Nachricht symbolweise abgearbeitet.

Im ersten Schritt wird das Intervall [0,1] in Intervalle I_1, ..., I_n unterteilt wobei n die Anzahl im Alphabet vorkommender Symbole ist und die Länge von I_k gleich der Symbolwahrscheinlichkeit des Symbols k ist. Durch das erste Symbol wird ein solches Intervall ausgewählt.

Die Prozedur beginnt nun mit dem zweiten Symbol der Nachricht und dem Restintervall anstelle von [0,1] von vorne.

Dieser Schritt wird solange wiederholt bis die gesamte Nachricht abgearbeitet ist.



Um die Nachricht zu übertragen genügt es die Intervallgrenzen in Binärdarstellung zu übertragen. Ist die Länge der Nachricht bekannt, so genügt es sogar eine beliebige Zahl im Intervall zu übertragen.

Adaptive Arithmetische Kodierung

Die Kodierung kann noch verbessert werden, wenn die Symbolwahrscheinlichkeiten für das nächste Symbol nicht statisch festgelegt sind, sondern von einem Modell geschätzt werden.

Ein solches Modell kann die Nachrichtensymbole S_0, ..., S_{i-1} benutzen um die Symbolwahrscheinlichkeiten für das i-te Symbol zu berechnen. Auf diese Weise kann sich daß Modell an eine Nachricht anpassen. Man spricht dann von adaptiver arithmetischer Kodierung.

Das einfachste solche Modell ist ein Histogramm. Es besitzt einen Zähler für jedes Symbol und schätzt die Symbolwahrscheinlichkeiten als Zählerstand geteilt durch die Summe aller Zähler. Die Zähler werden mit 1 initialisiert. Überschreitet die Summe einen Schwellwert, so werden alle Zähler inkrementiert und durch zwei geteilt.

Je kleiner diese Schwelle gewählt wird, um so schneller adaptiert das Modell; allerdings limitiert der Quotient

(Schwelle - Größe des Alphabets) / Schwelle

die größtmögliche Symbolwahrscheinlichkeit die das Modell vorhersagen kann. Tritt ein Symbol mit größerer Wahrscheinlichkeit auf so ist die Kodierung suboptimal.

Das EZW-Verfahren verwendet im subordinate-pass ein Histogramm mit zwei Zählern für die Symbole 0,1. Im dominant-pass verwendet es vier unabhängige Histogramme welche je einen der vier Fälle aus der Kombination

Vater signifikant / nicht signifikant
vorheriger Koeffizient signifikant / nicht signifikant

abdecken. Der Schwellwert wurde jeweils gleich 256 gewählt.

Zusammenfassung

Zum Abschluß die wichtigsten Eigenschaften des EZW-Verfahrens noch einmal zusammengefaßt.

Das Verfahren beinhaltet keine statistischen Resultate und muß nicht trainiert werden.

Der gewählte QMF-Filter (9-tap symmetrisch) liefert eine nahezu orthonormale Matrix wodurch nur ein Schwellwert für alle Koeffizienten benötigt wird.

Die Zerotree-Darstellung liefert in Verbindung mit der QMF-Pyramide und der arithmetischen Kodierung sehr befriedigende Resultate (meistens besser als JPEG).

Es entsteht bei gleicher Datenrate kein auffälliger Bildqualitätsverlust durch das eingebettete Verfahren.

Es entstehen auch bei sehr geringen Datenraten keine störenden Blockartefakte.

Das Prioritätsprotokoll erlaubt eine progressive sehr gleichmäßige Bildaufbereitung ohne den 'Scheibenwischer-Effekt' von progressive JPEG.