7 Bildkompression und Dateiformate

Digitale Einzelbilder und insbesondere Bildsequenzen (z.B. Filme) benötigen in der Regel sehr viel Speicherplatz und entsprechend lange Übertragungszeiten (z.B. im Internet). In den folgenden Teilkapiteln werden die gebräuchlichsten Ansätze zur Reduzierung des erforderlichen Speicherbedarfs vorgestellt.

Bei Rasterbilddaten geschieht dies (häufig unter Inkaufnahme von Qualitätsverlusten) durch unterschiedliche Bildkompressionsverfahren; siehe Kapitel 7.1. Bei Bildern, die ursprünglich nicht gerastert vorliegen und die z.B. mit Hilfe eines Zeichen- oder Konstruktionsprogramms erstellt wurden, speichert man statt des gerasterten Bildes die vektorielle Beschreibung des Bildes ab; siehe Kapitel 7.3. In Kapitel 7.2 werden außerdem die wichtigsten Rasterbild-Dateiformate vorgestellt; die gängigsten bildbeschreibenden Dateiformate werden in Kapitel 7.2 behandelt.

7.1 Kompressionstechniken für Rasterbilddaten

Das wichtigste Unterscheidungsmerkmal von Rasterbild-Kompressionsverfahren ist, ob sie eine verlustfreie oder verlustbehaftete Rekonstruktion der Bilddaten zulassen. Während bei den Verfahren mit ***verlustfreier Rekonstruktion*** das rekonstruierte Bild exakt mit dem Originalbild übereinstimmt, verzichtet man bei den Verfahren mit ***verlustbehafteter Rekonstruktion*** auf "unwichtige" Details. Dabei kann man i.d.R. die gewünschte Rekonstruktions-Qualität wählen und damit die Größe der komprimierten Bilddatei beeinflussen.

Bei Kompressionsverfahren unterscheidet man weiter zwischen ***Redundanz-*** und ***Irrelevanzreduzierenden Verfahren***. (Redundanzreduzierende Kompressionen werden auch als ***Entropie-Kodierung*** bezeichnet). Unter Redundanz versteht man dabei, dass Informationen aufgrund räumlicher oder zeitlicher Ähnlichkeiten in den Daten mehrfach vorhanden sind; (z.B. homogene Bildbereiche in Einzelbildern bzw. ähnliche aufeinanderfolgende Bilder einer Bildfolge). Irrelevanz bezieht sich auf Informationsanteile, die auf der Empfängerseite im Hinblick auf eine geforderte Rekonstruktionsgüte unerheblich sind; (z.B. sehr feine und kaum wahrnehmbare Bilddetails).

Eine weitere Unterscheidung unterteilt die Codierungsverfahren in ***eindimensionale*** und ***zweidimensionale***, meist ***blockorientierte Verfahren***. Eindimensionale Verfahren können nur eindimensionale Daten (z.B. Strings) verarbeiten, so dass Bilddaten zuerst in eine geeignete eindimensionale Anordnung überführt werden müssen. Hierzu kann man z.B. die Zeilen eines Bildes linear hintereinander anordnen. Bei den blockorientierten Verfahren werden die zweidimensionalen Eingangsdaten in eine gitterartige Anordnung rechteckiger Blöcke zerlegt, von denen jeder isoliert betrachtet wird. Daneben gibt es noch die ***globalen Verfahren***, bei denen das Bild als Ganzes betrachtet und komprimiert wird.

Im Vergleich der Zeiten für die Codierung bzw. Decodierung unterscheidet man zwischen ***symmetrischen*** und ***asymmetrischen Verfahren***. Bei den symmetrischen Methoden benötigt die Codierung etwa genau soviel Zeit, wie die Decodierung. Im Gegensatz dazu dauert die Codierung bei den asymmetrische Methoden ein Vielfaches länger als die Decodierung.

Im Hinblick auf die Decodierungsansätze wird zwischen ***sequentieller*** und ***progressiver Decodierung*** unterschieden. Bei der sequentiellen Decodierung wird ein Bild Zeile für Zeile auf dem Bildschirm aufgebaut. Im Gegensatz dazu zeigt die progressive Decodierung gleich zu Beginn eine unscharfe bzw. vergröberte Kopie des ganzen Bildes an, die dann schrittweise verbessert wird. Auf diese Weise kann man z.B. bei Web-Anwendungen die Übertragung eines Bildes frühzeitig abbrechen und so teure Übertragungszeit und Speicherplatz einsparen, wenn es sich nicht um das gewünschte Bild handelt. Wenn ein Bild in mehreren Auflösungsstufen verfügbar ist, so spricht man von ***hierarchischer*** oder ***skalierbarer Codierung***.

Im folgenden werden die drei für Bilddaten am häufigsten eingesetzten Kompressionstechniken kurz erläutert: Lauflängencodierung, Huffman Codierung und LZW-Codierung.

7.1.1 Lauflängencodierung (Run Length Encoding RLE)

Die ***Lauflängencodierung (Run Length Encoding, RLE)*** stellt eine eindimensionale Kompressionstechnik dar, bei der die in homogenen Bildbereichen enthaltenen Redundanzen ausgenutzt werden; (räumliche Kohärenz).

Bei der Lauflängencodierung wird in der eindimensionalen Anordnung der Farbwerte des Bildes (siehe oben) jedes Intervall identischer Werte dadurch ersetzt, dass man den Wert selbst und die Häufigkeit seines Auftretens im Intervall angibt.

Da bei der Lauflängencodierung keine Bildinformationen unterdrückt werden, ist eine verlustfreie Rekonstruktion der Bilder möglich. Die Lauflängencodierung, die zu den symmetrischen Ansätzen zählt, empfiehlt sich jedoch nur, wenn größere homogen eingefärbte Bildbereiche vorhanden sind.

7.1.2 LZW-Codierung

Die ***LZW-Codierung*** nach **L**empel, **Z**iv und **W**elch ist eine eindimensionale Kompressionstechnik, die davon ausgeht, dass die eindimensionale Anordnung der Farbwerte Abschnitte enthält, die an mehreren verschiedenen Stellen unverändert auftreten. An allen Stellen, an denen ein bereits bekannter Abschnitt auftritt, wird statt dieses Abschnitts nur ein Rückverweis auf das erste Auftreten dieses Abschnittes abgespeichert.

Die LZW-Codierung stellt eine musterorientierte Kompressionstechnik dar, die eine verlustfreie Rekonstruktion ermöglicht. Mit der LZW-Codierung, die zu den redundanzreduzierenden, symmetrischen Verfahren zählt, kann man für ein Bild nur dann eine erfolgreiche, d.h. platzsparende Codierung durchführen, wenn das Bild große homogene Bereiche oder sich häufig wiederholende Muster, wie z.B. Texturen, besitzt.

7.1.3 Huffman-Codierung

Der auf Huffman zurückgehenden ***Huffman-Codierung***, die ebenfalls ein eindimensionales Codierungsverfahren ist, liegt dieselbe Idee wie dem Morsealphabet zugrunde. Zunächst wird für alle auftretenden Farbwerte die Häufigkeit ihres Auftretens bestimmt. Anschließend wird ein Code variabler Länge verwendet, um häufig auftretende Farbwerte mit einem sehr kurzen Code, seltener auftretende Farbwerte dagegen mit einem längeren Code zu repräsentieren.

Die Huffman-Codierung gehört zu den verlustfreien Kompressionsverfahren. Sie empfiehlt sich nur, wenn die Häufigkeitsverteilung der zu codierenden Werte deutlich von einer Gleichverteilung abweicht. Bei realen Bilddaten ist der Kompressionsfaktor i.d.R. niedrig. Die Huffman-Codierung wird daher in der Praxis erst dann auf Bilddaten angewandt, wenn diese zuvor einer speziellen Transformation unterzogen wurden, die eine klare Ungleichverteilung der Werte bewirkt; (z.B. Diskrete Cosinus-Transformation bei der JPEG-Kompression; siehe Kapitel 7.1.5).

7.1.4 Fraktale Bildkompression

Die ***fraktale Bildkompression*** wurde aus der fraktalen Geometrie heraus entwickelt, die ihrerseits ein Teilgebiet der Chaostheorie ist.

Das Grundprinzip der fraktalen Kompression basiert auf der sogenannten ***Selbstähnlichkeit***, die in der Natur genauso wie in realen fotografischen Bildern überall zu beobachten ist. Der Begriff „Selbstähnlichkeit“ sagt hier aus, dass viele Naturphänomene im Großen wie im Kleinen dieselben Charakteristika aufweisen: die Küstenlinie eines ganzen Kontinentes zeigt im Satellitenbild dieselbe Struktur von Ein- und Ausbuchtungen, wie ein kleiner Küstenabschnitt von wenigen Kilometern oder Metern Länge; die Hauptäste eines Laubbaumes sehen wie verkleinerte Kopien des ganzen Baumes aus und dasselbe gilt auch für die kleineren Äste des Baumes.

Diese Selbstähnlichkeit ist die Grundidee des sogenannten ***Collage-Theorems*** [Barn], demzufolge sich eine natürliche Figur aus geeignet geometrisch transformierten (d.h. verkleinerten, verzerrten, gedrehten und verschobenen) Kopien der Figur selbst zusammensetzen läßt.

Mit Hilfe des Collage-Theorems kann man in der Computer Graphik mit geringem Konstruktionsaufwand und minimalem Speicherbedarf realistisch wirkende Objekte und Objektoberflächen wie z.B. Pflanzen, Planeten oder Landschaftsausschnitte erzeugen; beispielsweise lässt sich ein realistisch aussehendes Schwarzweißbild eines Farns, das obendrein beliebig vergrößerbar ist, mit Hilfe von 24 leicht zu erzeugenden reellen Zahlen erstellen.

Das Collage-Theorem lässt sich aber nicht nur zur Erzeugung synthetischer Bilder, sondern auch zur effizienten Bildkompression einsetzen. So wird bei der fraktalen Kompression eines Bildes nach Bereichen gesucht, die an anderen Stellen des Bildes nochmals in geeignet geometrisch transformierter Form auftreten; (z.B. Verkleinerungen, Scherungen und Rotationen). Auch photometrische Änderungen (wie Farb-, Helligkeits- oder Kontraständerungen) sind hierbei zugelassen.

Wenn man das ganze Bild mit derartigen Bereichen abdeckt, so genügt es, sich die Parameter der verwendeten geometrischen und photometrischen Transformationen zu merken, um das Bild zu rekonstruieren.

Da in der Regel einige tausend Transformationen ausreichen und diese durch wenige Parameter beschrieben werden, erreichen fraktale Kompressionsverfahren oft extreme Kompressionsraten mit zum Teil weniger als fünf Prozent des ursprünglichen Speicherbedarfs.

Die fraktale Kompression, die mit Redundanz- und Irrelevanz-Reduktion arbeitet, gehört zu den verlustbehafteten Verfahren, jedoch ist der Qualitätsverlust meist relativ gering. Während sehr feine und gering kontrastierende Bilddetails oft ungenau rekonstruiert werden, werden starke Kontrastsprünge im Gegensatz zu anderen verlustbehafteten Verfahren weder abgeschwächt noch verwischt. Ein weiterer Vorteil ist, dass ein fraktal komprimiertes Bild in beliebiger Auflösung rekonstruiert werden kann, und dass Bildausschnitte beliebig vergrößerbar sind, ohne dass blockartige Pixelstrukturen auftreten.

Die fraktale Kompression, die zu den zweidimensionalen Kompressionsverfahren gehört, ist aufgrund der Vielzahl zu untersuchender Transformationen sehr rechenintensiv. Im Gegensatz dazu konvergiert die Dekompression, bei der alle ermittelten Transformationen iterativ immer wieder angewendet werden, i.d.R. in wenigen Schritten und mit geringem Rechenaufwand, so dass es sich um ein asymmetrisches Verfahren handelt. Die Dekompression erfolgt häufig progressiv.

7.1.5 Diskrete Cosinus-Transformation mit Frequenzfilterung

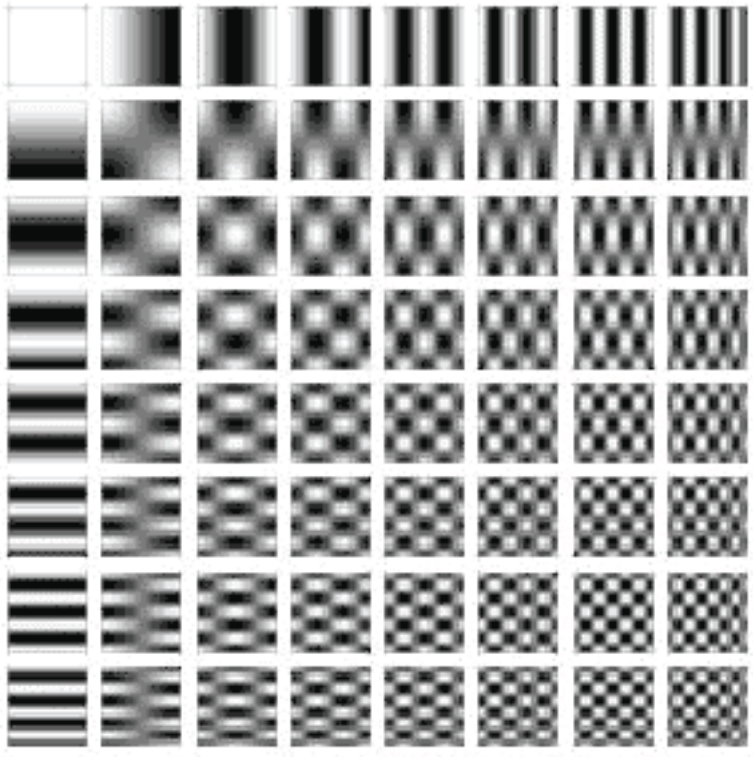
Die ***Diskrete Cosinus-Transformation*** (DCT) ist eine bei der Kompression zwei- und dreidimensionaler Daten (z.B. konventionelle 2D-Bilder und 3D-Volumenbilder) sehr häufig eingesetzte Transformation. Mit Hilfe der DCT, die eng mit der Diskreten Fourier-Transformation verwandt ist, werden die Eingangsdaten vom Orts- in den Ortsfrequenzbereich transformiert. Die eigentliche Kompression erfolgt im Ortsfrequenzbereich. Dabei behält man die besonders informationsträchtigen niedrigen Frequenzanteile bei, reduziert aber die hohen Frequenzen, die nicht oder kaum wahrnehmbaren Bilddetails entsprechen; (da das Auge bei hohen Frequenzen nicht so empfindlich auf Frequenzunterschiede reagiert, können diese gröber quantisiert werden). Durch eine inverse Diskrete Cosinus-Transformation können die Ausgangsdaten mit geringen Verlusten rekonstruiert werden.

Bei der DCT wird eine Bildmatrix zunächst in Blöcke von je 8*8-Pixel zerlegt. Jeder dieser Blöcke wird anschließend getrennt transformiert. Die Grundidee der Transformation besteht darin, dass man jeden Block als gewichtete Summe der in Abbildung 7.1 (a) gezeigten 64 Basisfunktionen darstellt. Dies entspricht einer Überlagerung der passenden Basisfunktionen mit entsprechend gewählten Amplituden. Die Matrix dieser 8*8 reellwertigen Amplituden (inkl.

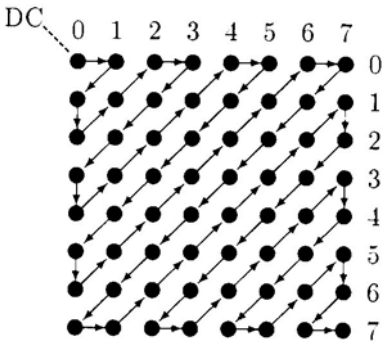
Vorzeichen) wird als Koeffizientenmatrix C(u,v) bezeichnet und stellt das Ergebnis der DCT dar:

$$C(u,v) = \frac{1}{4} * c(u) * c(v) * \sum_{j=0}^{7} \sum_{i=0}^{7} \left(g(i,j) * \cos\left(\frac{(2i+1)u\eth}{16} \right) * \cos\left(\frac{(2j+1)v\eth}{16} \right) \right)$$

Dabei steht g(i,j) für den Grauwert in Spalte i und Zeile j des jeweils zu transformierenden 8*8-Blockes, C(u,v) ist der resultierende Koeffizient (Amplitude) der Basisfunktion in der u. Spalte und v. Zeile aus Abbildung 7.1(a). Die Faktoren c(u) und c(v) haben den Wert $1/\sqrt{2}$, falls u bzw. v Null sind; ansonsten gilt c(u)=1 bzw. c(v)=1.



(a)



(b)

Abb. 7.1: (a) Basisfunktionen der Diskreten Cosinus-Transformation.
(b) Zick-Zack-Kurs zur eindimensionalen Abarbeitung eines Bildblockes.

Das eigentliche Vorgehen der DCT wird mit Hilfe des folgenden Beispiels demonstriert; ([Reim], Seite 62).

Beispiel:

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

(a)

235,6	– 1.0	– 12.1	– 5.2	2.1	– 1.7	– 2.7	– 1.3
– 22.6	– 17.5	– 6.2	– 3.2	– 2.9	– 0.1	0.4	– 1.2
– 10.9	– 9.3	– 1.6	1.5	0.2	– 0.9	– 0.6	– 0.1
– 7.1	– 1.9	0.2	1.5	0.9	– 0.1	0.0	0.3
– 0.6	– 0.8	1.5	1.6	– 0.1	– 0.7	0.6	1.3
– 1.8	– 0.2	1.6	– 0.3	– 0.8	1.5	1.0	– 1.0
– 1.3	– 0.4	– 0.3	– 1.5	– 0.5	1.7	1.1	– 0.8
– 2.6	1.6	– 3.8	– 1.8	1.9	1.2	– 0.6	– 0.4

(b)

Abb. 7.2: Beispiel zur Diskreten Cosinus-Transformation
(a) 8*8-Block (Bildausschnitt) (b) DCT-Koeffizienten

Das Beispiel zeigt die typische Wirkungsweise der DCT an einem 8*8 Pixel großen Bildausschnitt (Abbildung 7.2 (a)): nach der DCT sind in der Koeffizientenmatrix (Abbildung 7.2 (b)) nur noch wenige betragsmäßig große Werte vorhanden, und diese stehen vorwiegend in

der linken, oberen Ecke, die den niedrigen Frequenzanteilen des Bildes entspricht. Der Wert in der linken oberen Ecke, der dem konstant eingefärbten Feld in Abbildung 7.1 (a) entspricht, wird als Gleichspannungs- bzw. DC-Koeffizient bezeichnet. Alle anderen Koeffizienten sind die Wechselspannungs.- bzw. AC-Koeffizienten.

Im nächsten Schritt erfolgt nun mittels Frequenzfilterung die eigentliche Datenreduktion, indem alle Koeffizienten mit Hilfe der sogenannten Normalisierungs- oder Quantisierungstabelle gedämpft werden. In der unten gezeigten Fortsetzung des Beispiels ist eine Quantisierungstabelle gezeigt (Abbildung 7.3 (a)), wie sie vom JPEG-Komitee für Grauwertbilder bzw. für die Luminanzkomponente von Farbbildern empfohlen wird. Jeder DCT-Koeffizient wird durch den an derselben Position stehenden Wert der Quantisierungstabelle dividiert und das Ergebnis wird zum normalisierten DCT-Koeffizienten gerundet.

Fortsetzung des Beispiels:

(a)

16	11	10	16	24	40	51	61
12	12	143	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
25	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

(b)

15	0	- 1	0	0	0	0	0
- 2	- 1	0	0	0	0	0	0
- 1	- 1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(c)

240	0	- 10	0	0	0	0	0
- 24	- 12	0	0	0	0	0	0
- 14	- 13	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(d)

144	146	149	152	154	156	156	156
148	150	152	154	156	156	156	156
155	156	157	158	158	157	156	155
160	161	161	162	161	160	157	155
163	163	164	163	162	160	158	156
163	164	164	164	162	160	158	157
160	161	162	162	162	161	159	158
158	159	161	161	162	161	159	158

Abb. 7.3: Beispiel zur Diskreten Cosinus-Transformation (Fortsetzung)
(a) Quantisierungstabelle (b) normalisierte DCT-Koeffizienten
(c) rekonstruierte DCT-Koeffizienten (d) rekonstruierter Bildausschnitt

Das Beispiel zeigt in Abbildung 7.3 (b), dass in der Matrix der normalisierten DCT-Koeffizienten nur noch wenige Werte im linken oberen Bereich, der den niedrigen Frequenzanteilen entspricht, von Null verschieden sind.

Im nächsten Schritt wird für die AC-Koeffizienten dieser Matrix eine Lauflängencodierung vorgenommen, wobei wegen der speziellen Belegung der Matrix der Zick-Zack-Kurs aus Abbildung 7.1 (b) zur Umwandlung der 2D-Matrix in eine 1D-Zeichenkette verwendet wird. Für den DC-Koeffizienten wird die Differenz zum DC-Koeffizienten des vorangegangenen 8*8-Blockes abgespeichert. In einem letzten Schritt werden die DC-Differenz und das Ergebnis der Lauflängencodierung durch die Verwendung einer Huffman-Codierung nochmals (verlustfrei) komprimiert.

Auf diese Weise kann der im Beispiel gezeigte 8*8-Bildblock mit nur 31 Bit repräsentiert werden. Indem alle diese Schritte in umgekehrter Reihenfolge durchlaufen werden, kann das Bild mit wenigen Verlusten rekonstruiert werden; siehe Abbildungen 7.3 (c) und (d). Die verlustfreie inverse DCT ist dabei gegeben durch:

$$g(i,j) = \frac{1}{4} * \sum_{v=0}^{7} \sum_{u=0}^{7} \left(C(u,v) * c(u) * \cos\left(\frac{(2i+1)u\eth}{16} \right) * c(v) * \cos\left(\frac{(2j+1)v\eth}{16} \right) \right)$$

(Die Bedeutung von C(u,v), g(i,j) sowie c(u) und c(v) ist dabei wie auf der vorherigen Seite angegeben).

Bei der Diskreten Cosinus-Transformation (DCT) mit nachfolgender Frequenzfilterung, die ein sehr häufig eingesetztes verlustbehaftetes Kompressionsverfahren darstellt (z.B. JPEG- und MPEG-Codierung), handelt es sich um ein symmetrisches Verfahren.

Die DCT mit nachfolgender Frequenzfilterung erreicht sehr hohe Kompressionsraten, jedoch können hochfrequente Bilddetails verloren gehen und starke Kontrastsprünge abgeschwächt oder verwischt werden. Außerdem werden u.U. bei hohen Kompressionsraten die Grenzen der 8*8-Blöcke sichtbar.

7.1.6 Wavelet-Transformation mit Quantisierung

Die ***Wavelet-Transformation*** wird im Bereich der Computer Graphik hauptsächlich zur Bildkompression eingesetzt. Gegenüber der Diskreten Cosinus-Transformation mit Frequenzfilterung erzielt die Wavelet-Transformation mit Quantisierung insbesondere bei hohen Kompressionsraten qualitativ deutlich bessere Ergebnisse.

Die Wavelet-Transformation analysiert ein Bild in unterschiedlichen Auflösungsstufen (Bildpyramide), wobei im weiteren zur Vereinfachung davon ausgegangen wird, dass das Bild quadratisch und seine Kantenlänge eine Zweierpotenz ist. Das Grundprinzip, das iterativ immer wieder angewendet wird, ist einfach: für eine Auflösungsstufe eines Bildes wird zunächst eine zeilenweise Glättung (Tiefpass-Filterung) sowie eine Reduzierung der Bildbreite auf die Hälfte vorgenommen; siehe Abbildung 7.4 oben links. Die bei der Glättung verloren gegangenen Details werden in einem zweiten Bild erfasst, das ebenfalls auf die Hälfte der ursprünglichen Bildbreite verkleinert wird; siehe Abbildung 7.4 unten links. Da die Verluste bei der Glättung genau an den vertikalen Bildkanten auftreten, entspricht die Erzeugung des Detailbilds der Anwendung eines Kantendetektors für senkrechte Kanten (gerichtete Hochpass-Filterung). In einem weiteren Schritt werden die beiden Teilbilder nun spaltenweise geglättet und auf die halbe Höhe reduziert und die dabei entstehenden Detail-Verluste werden wieder in Detailbildern erfasst; siehe rechte Seite von Abbildung 7.4.

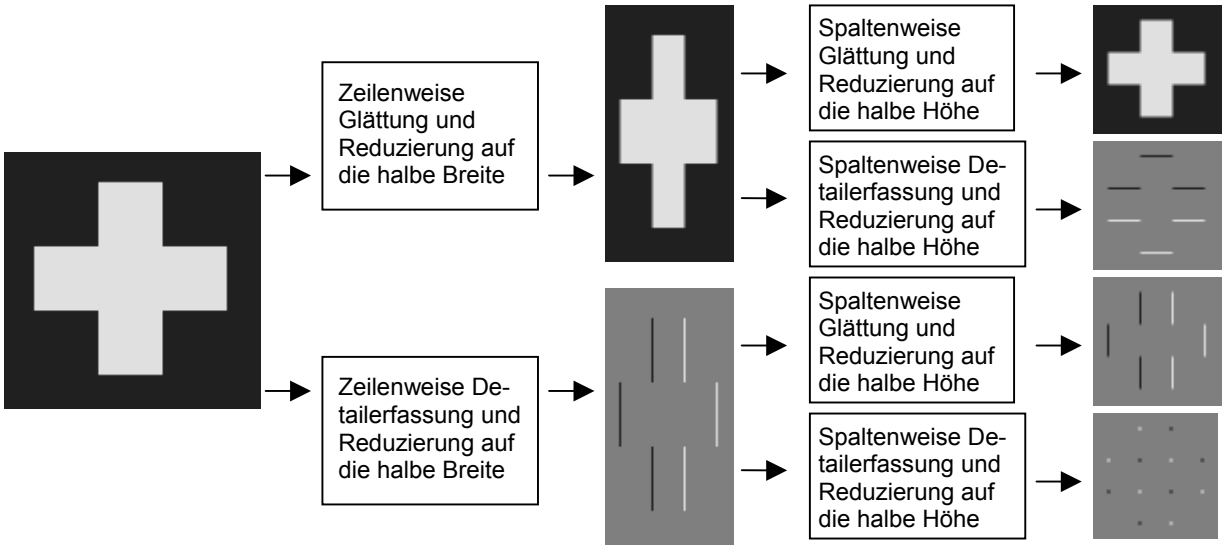
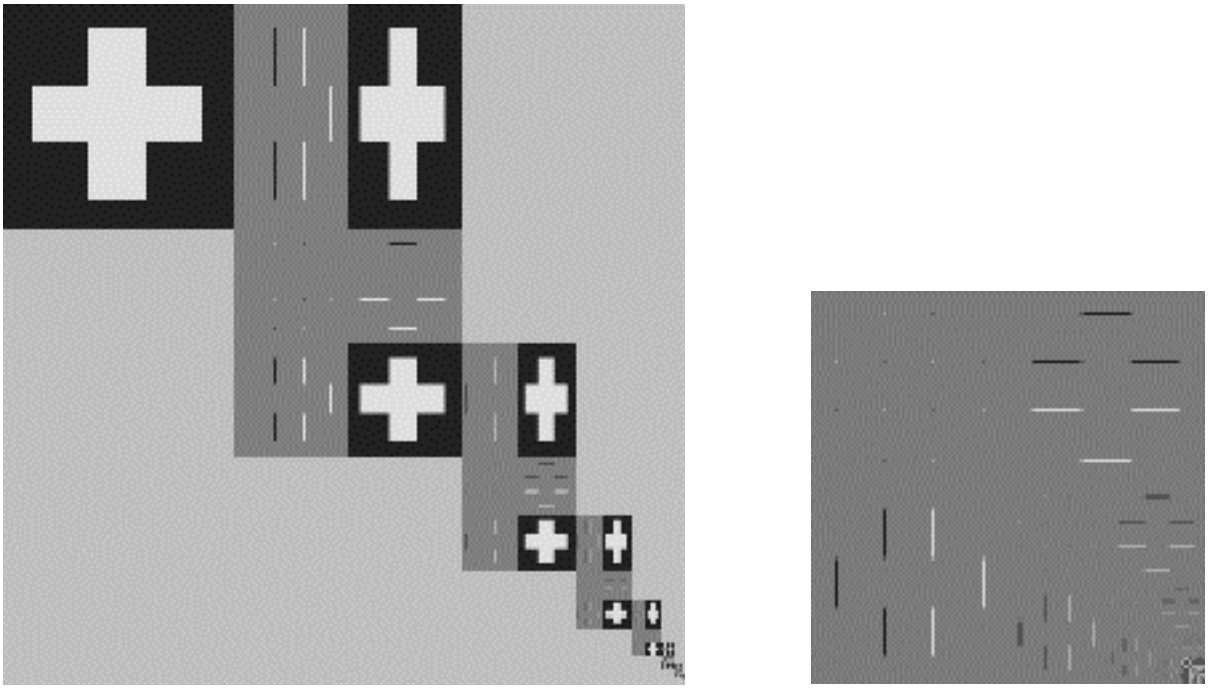


Abb. 7.4: Beispiel für einen Schritt der Wavelet-Transformation; die Farbe Mittelgrau in den Detailbildern entspricht dem Wert Null.

Die drei Detailbilder, die jeweils ein Viertel der Größe des ursprünglichen Bildes besitzen, enthalten die sogenannten ***Wavelet-Koeffizienten*** dieses Iterationsschrittes. Für das ebenfalls in beide Richtungen um den Faktor zwei verkleinerte geglättete Bild wird nun dasselbe Vorgehen aus Tief- und Hochpass-Filterung wiederholt. Bei jedem Iterationsschritt werden zunehmend gröbere Bildstrukturen herausgefiltert und das letzte, ein Pixel große Bild zeigt den mittleren Grauwert des Originalbildes. In Abbildung 7.5 (a) ist das iterative Vorgehen bei der Wavelet-Transformation vom Originalbild (oben links) bis zum letzen Filterungsschritt (unten rechts) dargestellt. Die vier Teilbilder aus Abbildung 7.4 sind oben links im diagonal unter dem Originalbild stehenden Bild zu sehen.

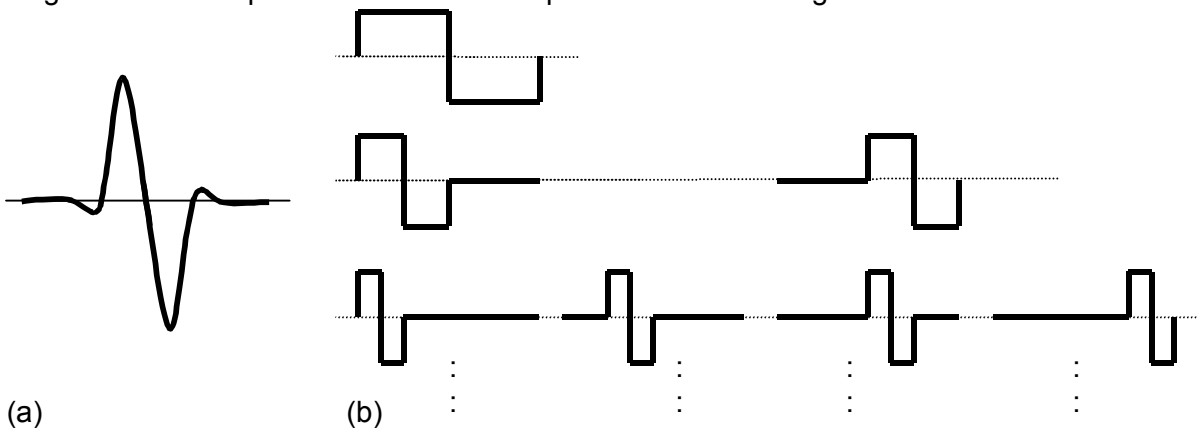


(a) (b)

Abb. 7.5: Beispiel für das iterative Vorgehen bei der Wavelet-Transformation; die Farbe Mittelgrau in den Detailbildern entspricht dem Wert Null.

Die aus allen Iterationsschritten angesammelten Wavelet-Koeffizienten bilden zusammen mit dem auf ein Pixel reduzierten Ausgangsbild das Endergebnis der Wavelet-Transformation; siehe Abbildung 7.5 (b). Diese Koeffizienten-Matrix, die die verlustfreie Rekonstruktion des Originalbildes erlaubt, hat dieselben Abmessungen, wie das Originalbild. Entscheidend ist aber, dass die meisten Koeffizienten Null oder aber betragsmäßig sehr klein sind.

Da betragsmäßig kleine Wavelet-Koeffizienten nur geringe Grauwert-Änderungen bei der Rekonstruktion des Bildes bewirken, kann man diese im Rahmen einer Irrelevanz-Reduktion unterdrücken. Dabei ist allerdings zu bedenken, dass die bei den iterativen Verkleinerungen ermittelten Koeffizienten von Schritt zu Schritt wichtiger werden, da sie einen größeren Bereich des Bildes betreffen. Die Unterdrückung der Wavelet-Koeffizienten wird durch eine spezielle ***Quantisierung*** erreicht, die unterschiedliche Bereiche der Koeffizienten-Matrix unterschiedlich grob quantisiert. Mit Hilfe dieser Quantisierung kann auch die Rekonstruktionsgüte bzw. der Speicherbedarf des komprimierten Bildes eingestellt werden.



(a) (b)

Abb. 7.6: (a) Beispiel für ein MotherWavelet;
(b) drei Auflösungsstufen aus der Haar-Wavelet-Basis.

Für die Tief- und Hochpass-Filterung stehen unterschiedliche Mother-Wavelets zur Verfügung, wobei man unter einem ***Wavelet*** (englisch für "kleine Welle") eine besondere Wellen-

form versteht, die (z.B. im Gegensatz zur Cosinus-Funktion) nur in einem eng begrenzten Intervall von Null verschieden ist. Abbildung 7.6 (a) zeigt ein spezielles Mother-Wavelet. Aus dem Mother-Wavelet werden iterativ verkleinerte und gegenseitig verschobene Versionen erzeugt, die in ihrer Gesamtheit als ***Wavelet-Basis*** bezeichnet werden. Abbildung 7.6 (b) zeigt einen Teil der diskreten Haar-Wavelet-Basis. Bei der iterativen Tief- und Hochpass-Filterung des Originalbildes werden von Schritt zu Schritt die nächst "größeren" Versionen der Wavelets verwendet.

Die Wavelet-Transformation mit nachfolgender irrelevanzreduzierender Quantisierung, die zu den zweidimensionalen, globalen Kompressionsverfahren gehört, erzielt Ergebnisse, die hinsichtlich der Kompressionsrate und der Bildqualität mit der fraktalen Kompression vergleichbar sind. Im Gegensatz zur fraktalen Kompression handelt es sich hier aber um ein symmetrisches Verfahren, das mit vergleichsweise geringem Rechenaufwand auskommt.

7.2 Bilddatenformate

In diesem Teilkapitel werden die gängigsten Datenformate, die zur Archivierung und zum Austausch von Bilddaten verwendet werden, kurz vorgestellt. Neben den Einzelbildformaten BMP, PCX, GIF, PNG, TIFF, FIF und JPEG wird auch das Bildfolgenformat MPEG erläutert.

Zu den jeweiligen Formaten werden nur die für den Benutzer wesentlichen Aspekte (wie z.B. die Verwendung eines Kompressionsverfahrens und die Vor- und Nachteile des jeweiligen Formates) angesprochen. Auf die Schilderung des internen Aufbaus der jeweiligen Datenformate, der i.d.R. aus einem Vorspann (Header), dem eigentlichen Datenteil mit den Farbwerten und gegebenenfalls der Farbpalette, sowie u.U. weiteren Abschnitten besteht, wird hier nicht näher eingegangen. Einzelheiten können z.B. in [Born] oder [Lipp] nachgelesen werden.

7.2.1 BMP

Das Windows **Bi**t**ma**p Format BMP wurde von Microsoft anlässlich der Vorstellung von Windows 3.0 als geräteunabhängiges Datenformat für Rasterbilder eingeführt. Im BMP-Format können Bilder mit 2, 16, 256 oder ca. 16.7 Millionen Farben (True Color) verlustfrei abgespeichert werden. Dabei können Bilder mit 16 oder 256 Farben, die mit Hilfe einer Farbpalette definiert sind, nach dem RLE-Verfahren komprimiert werden. Das BMP-Format ist sehr einfach aufgebaut und wird direkt vom Windows-Graphikmodul GDI unterstützt.

Leider sind das Microsoft-BMP-Format und das von IBM ab OS/2 (2.x) verwendete BMP-Format nicht mehr kompatibel.

7.2.2 PCX

Das PCX-Format wurde ursprünglich von der ZSoft Corporation für die kompakte Speicherung und schnelle Übertragung von Graphiken, die mit dem Malprogramm Paintbrush erstellt wurden, entwickelt. Die weite Verbreitung von Paintbrush und die ständige Weiterentwicklung des verlustfreien PCX-Formates (z.B. für True-Color-Bilder) haben dazu geführt, dass das PCX-Format heute als gängiges Bilddatenformat etabliert ist. Zur Kompression wird das RLE-Verfahren eingesetzt.

7.2.3 GIF

Das **G**raphics **I**nterchange **F**ormat GIF wurde von CompuServe entwickelt und erlaubt neben der reinen Abspeicherung von Bilddaten z.B. die Anzeige von Texten in Rasterbildern und das Warten auf Benutzerreaktionen. Weiterhin können in einer GIF-Datei mehrere Bilder enthalten sein („animated GIF“), so dass kurze Animationen wie z.B. animierte Icons oder aus wenigen Bildern bestehende Spots effizient abgespeichert werden können. Für umfangreiche Bildfolgen ist GIF aufgrund der trotz Kompression recht großen Datenmenge nicht geeignet.

Bei der Entwicklung des HTML-Standards (**H**yper**t**ext **M**arkup **L**anguage) wurde GIF neben dem JPEG-Format als eine der Möglichkeiten zur Einbindung von Graphik in Internetseiten festgelegt.

GIF-Bilder können, obwohl sie rechteckig definiert sind, bei der Ausgabe beliebige Formen annehmen, was z.B. zur Gestaltung von Firmenschriftzügen und Logos genutzt werden kann. Dieser Effekt wird dadurch erreicht, dass Teile des rechteckigen Gesamtbildes als transparent markiert werden, so dass der Hintergrund, in den das GIF-Bild eingeblendet wird, an diesen Stellen durchscheint.

Ein Nachteil von GIF ist die Beschränkung auf maximal 256 verschiedene Farben, die in einer Farbtabelle erfasst werden. Zwar gibt es Verfahren, wie z.B. das Median-Cut- oder den Popularitäts-Algorithmus, die zu einem Bild mit beliebig vielen Farben eine optimal angepasste Farbtabelle mit nur noch 256 Farben berechnen, jedoch ist die Qualität des Ausgabebildes nicht so gut, wie bei einem Echtfarbbild. Außerdem ist der Platzbedarf eines derart farbreduzierten Bildes im Vergleich zu einer ebenfalls verlustbehafteten JPEG-Kompression (siehe Kapitel 7.2.7) in der Regel wesentlich höher.

Zur Komprimierung wird bei GIF die verlustfreie LZW-Codierung eingesetzt. Diese war bis vor kurzem patentrechtlich geschützt, so dass Software-Entwickler und Betreiber kommerzieller Internetseiten bei Verwendung von GIF-Bildern Lizenzgebühren bezahlen mussten. Mittlerweile darf die LZW-Codierung gebührenfrei genutzt werden.

GIF ist heute insbesondere im Bereich von Web-Anwendungen und dort besonders für synthetische Bilder bzw. Bilder mit einheitlich eingefärbten Objekten und Hintergründen immer noch ein sehr häufig eingesetztes Datenformat. Dies rührt daher, dass die i.d.R. deutlich reduzierte Bildgröße für den Anwender merklich kürzere Übertragungszeiten, d.h. geringere Telefonkosten und weniger Speicherbedarf mit sich bringt.

7.2.4 PNG

Das **P**ortable **N**etwork **G**raphics Format PNG (sprich „ping“) kann als Nachfolger von GIF angesehen werden. Statt der gebührenpflichtigen Verwendung des LZW-Verfahrens wird hier eine andere, vom ZIP-Format abgeleitete verlustfreie Kompressionstechnik eingesetzt. Außerdem erlaubt PNG auch die Speicherung von Echtfarbbildern. Da PNG im Vergleich zu GIF um bis zu 30 Prozent höhere Kompressionsraten erreicht, stellt PNG ab HTML 4.0 eine echte Alternative zu GIF für die Einbindung von Graphiken in Internetseiten dar.

Als zusätzliche Erweiterung zu GIF stellt PNG einen echten Alpha-Kanal (siehe Kapitel 3.2.2.1) zur Verfügung, mit dem für jedes Pixel der Grad der Transparenz eingestellt werden kann. Auf diese Weise lassen sich z.B. weiche Übergänge zwischen einem Hintergrundbild und dem eingeblendeten PNG-Bild erzielen.

PNG-Bilder können bereits während des Ladens in einer „blockigen“ Grobansicht angezeigt werden, die dann schrittweise verfeinert wird; (progressive Decodierung). Außerdem kann man in PNG-Dateien Stichworte z.B. zum Autor und zur Beschreibung des Bildes unterbringen, so dass Web-Benutzer mit entsprechender Software-Unterstützung beispielsweise gezielt nach Bildern mit bestimmten Bildinhalten suchen können.

Eine Erweiterung zu PNG ist das **MNG**-Format (**M**ultiple Image **N**etwork **G**raphics) geplant, mit dem Animationen abgespeichert werden können.

7.2.5 TIFF

Das **T**ag **I**mage **F**ile **F**ormat TIFF wurde ursprünglich von Aldus (Pagemaker) für Anwendungen im Druckbereich entwickelt. Aufgrund seiner ständigen Erweiterungen und Anpassungen hat sich TIFF mittlerweile als weitverbreitetes geräte- und betriebssystemunabhängiges Bilddateiformat etabliert.

Die Bezeichnung „Tag“ im Namen steht dabei für die Identifikationsnummern der separaten Bereiche des recht kompliziert aufgebauten TIF-Formats.

Neben den üblichen Schwarz-Weiß-, Grauton- und Farbbildern mit bis zu 16,7 Millionen Farben kann TIFF einen kompletten Alphakanal erfassen und Farbbilder können statt im RGB-Farbmodell auch im z.B. CMYK-Modell, im Video-Standbildformat YC_bC_r und im CIE-Format abgespeichert werden. Eine TIFF-Datei kann mehrere Bilder sowie Kommentare umfassen.

Zur Kompression stehen mehrere Alternativen zur Verfügung: z.B. modifizierte Versionen der Lauflängen-, Huffman- oder LZW-Komprimierung und seit einiger Zeit auch die JPEG-Komprimierung; siehe Kapitel 7.2.7.

Das TIF-Format ist eines der besonders flexiblen Formate. Aufgrund seines komplizierten Aufbaus arbeiten jedoch viele der verfügbaren Kompressions- und Dekompressionsverfahren nicht in allen Modi fehlerfrei.

7.2.6 FIF

Das **F**ractal **I**mage **F**ormat FIF wurde von der Firma Iterated Systems eingeführt, deren Mitbegründer M. Barnsley, der „Vater“ der fraktalen Bildkompression ist; siehe Kapitel 7.1.4.

FIF ist verlustbehaftet, d.h. das rekonstruierte Bild unterscheidet sich geringfügig vom Original. In der Kompressionsphase, die erheblich länger als die Dekompressionsphase dauert (asymmetrisches Verfahren), kann der Benutzer die Kompressionsrate und damit den Verlust an Detailgenauigkeit einstellen; die Kompression erfordert um so mehr Zeit und Speicherplatz je genauer die spätere Rekonstruktion möglich sein soll.

Erfahrungsgemäß kann man reale Bilder mit FIF bei kaum merklichen Qualitätsverlusten auf 5 bis 15 Prozent des ursprünglichen Speicherbedarfs reduzieren. Im Vergleich zu JPEG (s. Kap. 7.2.7) schneidet FIF bei photographischem Bildmaterial und gleich hoher Kompressionsrate in der Regel besser ab, da auch bei hoher Kompression keine Blockgrenzen sichtbar werden; siehe z.B. MS-Encarta.

Die Dekompression kann progressiv erfolgen, so dass dem Betrachter schon zu Anfang eine grobe Bilddarstellung gezeigt wird, die dann schrittweise verfeinert wird. Ein weiterer positiver Aspekt von FIF zeigt sich beim Vergrößern von Bildausschnitten. Das Zooming dauert zwar je nach Vergrößerungsstufe im Vergleich zu anderen Ansätzen erheblich länger, jedoch wirken die Ergebnisse deutlich besser, denn die fraktale Vergrößerungstechnik führt zu „glatt“ wirkenden Vergrößerungen, die nicht die üblichen quadratischen Pixelblöcke aufweisen.

Nachteilig an FIF ist, dass derzeit keine Transparenz (Alpha-Kanal) und keine Animationen vorgesehen sind. Eine Stichwortsuche, wie sie z.B. für Web-Benutzer interessant ist, wird ebenfalls noch nicht unterstützt.

7.2.7 JPEG

Das JPEG-Format wurde von der **J**oint **P**hotographic **E**xpert **G**roup als internationaler Standard zur Kompression von Schwarz-Weiß-, Graustufen und Farbbildern entwickelt. Nicht zuletzt, da JPEG neben GIF bei der Entwicklung des HTML-Standards als eine der Möglichkeiten zur Einbindung von Graphiken in Internetseiten festgelegt wurde, gehört JPEG heute wegen seiner hohen Kompressionsraten (insbesondere für Photographien) im Bereich des Internets zu den sehr häufig benutzten Bilddatenformaten. Mit JPEG können Bilddaten auf vier verschiedene Weisen codiert werden, von denen eine verlustfrei, die anderen aber verlustbehaftet arbeiten.

Die verlustfreie JPEG-Codierung verzichtet im Gegensatz zu den anderen drei Möglichkeiten auf eine Transformation in den Ortsfrequenzbereich und erreicht vergleichsweise

bescheidene Kompressionsraten. Diese Alternative wird deshalb im folgenden nicht weiter betrachtet.

Bei den übrigen drei Möglichkeiten wird die Diskrete Cosinus-Transformation (siehe Kapitel 7.1.5) mit nachfolgender Frequenzfilterung eingesetzt. Bei der einfachsten dieser Möglichkeiten, der sequentiellen Codierung, die auch als JPEG-Baseline bezeichnet wird, wird das zu codierende Bild systematisch von oben nach unten abgearbeitet. Entsprechend wird das Bild bei der Decodierung in voller Auflösung von oben nach unten aufgebaut.

Eine Erweiterung hierzu stellt die fortschreitende Codierung (progressive encoding) dar, bei der das Bild mehrfach bearbeitet wird, wobei in jedem Schritt die Bildqualität erhöht wird. Bei der Decodierung sieht der Benutzer also sehr früh ein vollständiges, aber unscharfes Bild und er kann den zeit- und kostenintensiven Bildaufbau bzw. die Bildübertragung gegebenenfalls frühzeitig abbrechen.

Die letzte Codierungsmöglichkeit ist die sogenannte hierarchische Codierung, bei der das Bild mehrfach in unterschiedlichen Auflösungen codiert wird (Bildpyramide). Auf diese Weise kann man z.B. auf eine niedriger aufgelöste Version eines Bildes zurückgreifen, ohne die höchstmögliche Auflösung decodieren zu müssen.

Im weiteren wird das Vorgehen bei der JPEG-Baseline Codierung erläutert, auf der die beiden anderen Ansätze aufbauen. Der Gesamtablauf mit seiner mehrfachen Komprimierung ist in Abbildung 7.7 schematisch dargestellt.

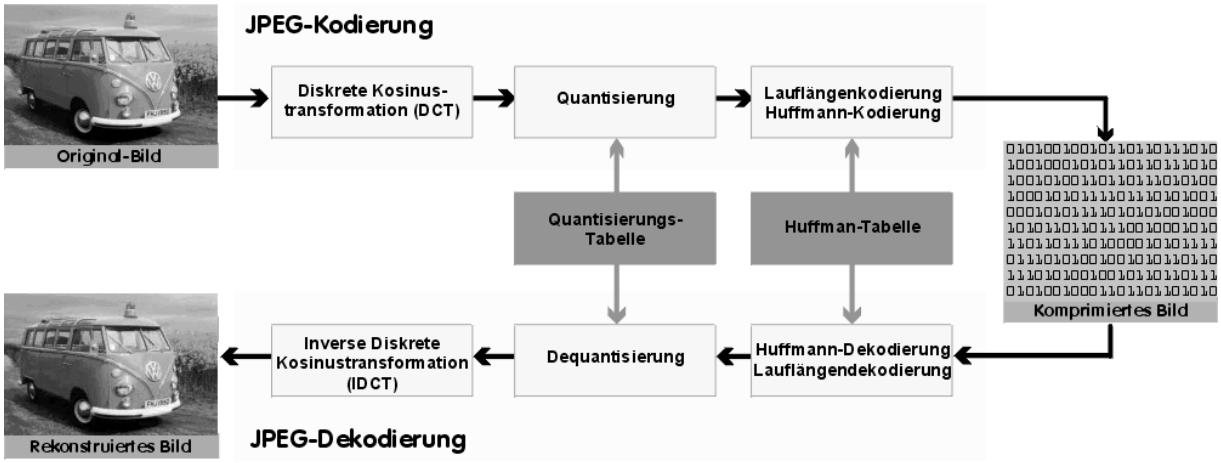


Abb. 7.7: Schematisches Vorgehen bei der JPEG-Baseline-Codierung

Die einzelnen Teilschritte aus Abbildung 7.7 wurden in Kapitel 7.1.5 kurz erläutert und werden hier nur noch einmal aufgezählt. Zunächst wird das Eingangsbild in quadratische Blöcke von 8*8 Bildpunkten zerlegt, die dann getrennt weiterverarbeitet werden. Für jeden Block wird eine FDCT (**F**ast **D**iscrete **C**osine **T**ransformation), die eine optimierte, schnelle Version der normalen DCT ist, durchgeführt, um die Bilddaten in den Ortsfrequenzbereich zu transformieren. Danach werden hohe Frequenzanteile, die nicht oder kaum wahrnehmbaren Bilddetails entsprechen, im Rahmen der Quantisierung unterdrückt; (1. Stufe der Kompression). Die verbleibenden Frequenzen werden anschließend in einem Zick-Zack-Kurs mit Hilfe einer Lauflängencodierung erfasst; (2. Stufe der Kompression). Das Ergebnis wird abschließend mit der lizenzgebührfreien Huffman-Codierung weiter komprimiert; (3. Stufe der Kompression). Die Decodierung dieses symmetrischen Ansatzes durchläuft alle Schritte in umgekehrter Reihenfolge.

Die JPEG-Codierung, die sich vor allem für photographische Aufnahmen hoher Auflösung und weniger für einfache Graphiken eignet, ist sehr zeitaufwendig, aber sie erlaubt wie FIF extrem hohe Kompressionsraten. Der Kompressionsfaktor, der sich umgekehrt proportional zur Rekonstruktionsgüte verhält, kann vom Benutzer fein abgestuft eingestellt werden. Für reale Bilder sind Kompressionen auf bis zu zehn Prozent des ursprünglichen Umfangs mög-

lich. Zu stark komprimierte Bilddaten neigen allerdings zu Rekonstruktionsfehlern, wie 8*8-Blockstrukturen, sowie zu „Schwingungen“ und Unschärfen an Kanten.

Schwarz-Weiß-Bilder und Grautonbilder können direkt verarbeitet werden, wohingegen Farbbilder, die eine Farbtabelle verwenden, zuerst in echte Farbbilder umgerechnet werden. Für Farbbilder wird i.d.R. der YUV-Farbraum verwendet, da sich hier eine weitere Kompressionsmöglichkeit durch Irrelevanz-Reduktion ergibt: während beim RGB-Farbmodell alle drei Farbkanäle für den Betrachter gleich wichtig sind, kann man beim YUV-Modell die Chrominanzen U und V mit der halben räumlichen Auflösung erfassen, ohne dass dies zu erkennbaren Qualitätsverlusten führt; (vgl. Kapitel 3.2.2.2).

Als Erweiterung zu JPEG wurde Motion-JPEG eingeführt, um auch Bildfolgen komprimieren zu können. Da hierbei aber die Ähnlichkeit der zeitlich aufeinanderfolgenden Einzelbilder nicht genutzt wird, schneidet Motion-JPEG im Vergleich zu MPEG (siehe 7.2.9) wesentlich schlechter ab und hat sich deshalb nicht durchsetzen können.

In naher Zukunft ist die Einführung von JPEG 2000 als neuem Standard geplant. Dabei wird anstelle der DCT die Wavelet-Kompression eingesetzt, die zu den globalen Verfahren gehört. JPEG 2000 verspricht bei gleichen Kompressionsraten eine bessere Bildqualität.

7.2.8 JPEG 2000

Wegen der Nachteile des auf der Diskreten Cosinus-Transformation aufbauenden JPEG-Formates, wurde das wavelet-basierte JPEG-2000-Format eingeführt.

Obwohl dieses Format gegenüber dem ursprünglichen JPEG bei gleicher Bildqualität deutlich kleinere Dateigrößen bzw. bei gleicher Kompressionsrate eine spürbare bessere Bildqualität gewährleistet, bleibt abzuwarten, ob sich JPEG-2000 auf dem Markt durchsetzt. Derzeit wird es nur von wenigen Browsern unterstützt und auch die Hersteller von Digital-Kameras setzen bei der Kompression der aufgenommenen Bilder meist noch auf das alte JPEG-Format.

Bei der JPEG-2000-Codierung eines Bildes schließt sich an die in Kapitel 7.1.4 geschilderte Wavelet-Transformation mit nachfolgender Quantisierung der Wavelet-Koeffizienten noch eine aufwendige verlustfreie Codierung der Koeffizienten an. Diese stützt sich insbesondere auf die Tatsache, dass in Abhängigkeit von der "Stärke" der eingestellten Quantisierung ein sehr hoher Anteil der Koeffizienten Null ist, und ansonsten nur noch wenige verschiedene Werte auftreten.

Durch eine geeignete Sortierung der quantisierten Koeffizienten kann außerdem erreicht werden, dass z.B. bei einer web-basierter Nutzung das Bild bereits nach wenigen übertragenen Daten in einer schlechten Auflösung angezeigt werden kann. Dazu müssen die Daten aus der rechten unteren Ecke der Koeffizientenmatrix zuerst übertragen werden. Indem die bei der Wavelet-Transformation vorgenommene iterative Verkleinerung des Bildes schrittweise rückgängig gemacht wird, entsteht das Bild mit der Übertragung der nächsten Detailbild-Koeffizienten Schritt für Schritt in immer besserer Auflösung (skalierbare progressive Übertragung und Decodierung).

JPEG-2000 ist in gleicher Weise für reale Bilder mit weichen Farbverläufen und für synthetische Bilddaten mit harten Farbübergängen geeignet. JPEG-2000 verhält sich robust gegen Übertragungsfehler und unterstützt Transparenz. Außerdem wird die Erfassung von Meta-Daten (z.B. zum Bildinhalt) unterstützt.

7.2.9 MPEG

Das MPEG-Format wurde von der **M**otion **P**icture **E**xpert **G**roup entwickelt und erlaubt, wie der Name aussagt, die Kompression von Bewegtbildern, d.h. Bildfolgen. Mit der MPEG-Kompression kann zusätzlich zu den Bilddaten auch Audio-Information wie z.B. die Tonspur eines Filmes erfasst werden. Der Audio-Aspekt, der auf einer eindimensionalen DCT beruht, wird im weiteren nicht betrachtet.

Während andere Verfahren wie z.B. GIF und JPEG dahingehend erweitert wurden, dass sie statt einem auch mehrere Bilder einzeln und unabhängig voneinander erfassen können (***Intraframe-Codierung***), wurde MPEG als ***Interframe-Codierungsverfahren*** entwickelt. Das bedeutet, dass zusätzlich zu den anderen Kompressionsansätzen noch die zeitliche Kohärenz, d.h. die Ähnlichkeit aufeinanderfolgender Bilder einer Bildfolge, ausgenutzt wird. Indem also im Prinzip nur die von Bild zu Bild auftretenden Änderungen abgespeichert werden, ergibt sich gegenüber der Abspeicherung der komprimierten Einzelbilder nochmals eine deutliche Datenreduktion.

Die Grundidee der bei MPEG eingesetzten Interframe-Codierung basiert darauf, dass für zwei zeitlich nicht zu weit auseinanderliegende Bilder B_1 und B_2 einer Bildfolge gilt, dass große Teile von B_1 in B_2 mehr oder weniger unverändert an derselben oder einer geringfügig verschobenen Bildposition wieder auftreten. Indem man die Verschiebungen für sogenannte Makroblöcke mit Hilfe von ***Verschiebevektoren*** angibt und die eventuellen Änderungen mit Hilfe von ***Differenzbildern*** beschreibt, lässt sich Bild B_2 mit wenigen Zusatzinformationen aus Bild B_1 rekonstruieren; siehe Abbildung 7.8.

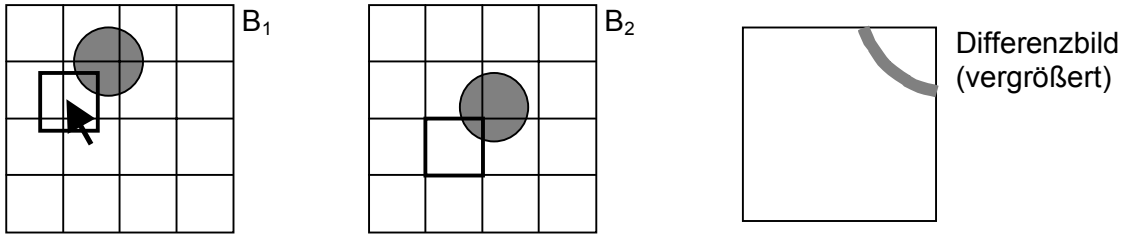


Abb. 7.8: Beispiel für die Ermittlung eines Verschiebevektors und eines Differenzbildes bei der MPEG-Codierung; der Ball ändert von Bild B_1 zu B_2 seine Position und Größe.
(a) Bild B_1 mit optimal verschobenem Makroblock aus Bild B_2 und zugehörigem Verschiebevektor;
(b) Bild B_2 mit hervorgehobenem Makroblock;
(c) Differenzbild (vergrößerte Darstellung).

Bild B_2 wird als ***unidirektional prädiziertes*** (vorhergesagtes) sogenanntes ***P-Bild*** bezeichnet. Ergänzend hierzu gibt es noch die ***bidirektionale Prädiktion***, bei der ein Bild aufgrund seiner Unterschiede zu einem vorhergehenden und einem nachfolgenden Bild vorhergesagt wird. Diese Bilder werden als ***B-Bilder*** bezeichnet.

Im konkreten Fall werden bei MPEG in bestimmten Abständen Bilder der Bildfolge vollständig mit der Intraframe-Codierung von JPEG als sogenannte ***I-Bilder*** abgespeichert. In den Bereichen zwischen diesen vollständig abgespeicherten I-Bildern werden in kleineren Abständen mehrere unidirektional prädizierte P-Bilder erfasst. Die restlichen Zwischenbilder werden bidirektional aus den vorangegangenen bzw. nachfolgenden I- und P-Bildern abgeleitet; siehe Abbildung 7.9.

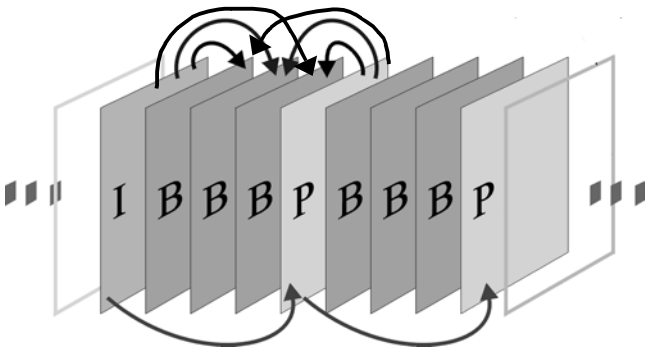


Abb. 7.9: Beispiel für die MPEG-Codierung eines Ausschnittes aus einer Bildfolge.

Der aufwendigste Teil der MPEG-Kompression ist die Bewegungsprädiktion, d.h. die Berechnung der Verschiebevektoren und Differenzbilder. Da dieser Aufwand bei der Decodie-

rung entfällt, handelt es sich bei MPEG um ein asymmetrisches Verfahren. Da die Abspeicherung bzw. Übertragung von Bildfolgen eine konstante Datenrate erfordert, muss die MPEG-Codierung kontinuierlich auf den Bildinhalt reagieren: z.B. würde ein fein strukturierter Bildinhalt mit vielen hohen Frequenzanteilen eine wesentlich gleichmäßiger besetzte Matrix der DCT-Koeffizienten liefern, als dies bei einem „ruhigen“ Bild der Fall wäre. Um in beiden Fällen eine Reduzierung auf die gleiche Datenmenge zu erreichen, wird im ersten Fall durch eine Anpassung der Quantisierungstabellen eine wesentlich gröbere Quantisierung der Frequenzen vorgenommen, was zu einer deutlich schlechteren Bildqualität führen kann.

Das Abspielen einer Bildfolge in umgekehrter Reihenfolge (Bildrücklauf), sowie der wahlfreie Zugriff auf Einzelbilder (z.B. zum Editieren und Schneiden) ist beim MPEG-Format aufgrund der Interframe-Codierung nur mit zusätzlichen Berechnungen und Zwischenspeicherungen möglich. Unproblematisch ist dagegen der schnelle Vor- und Rücklauf (Zeitraffer), da hier einfach nur auf die I-Bilder (und u.U. sogar nur auf die DC-Koeffizienten jedes 8*8-Blockes dieser Bilder) zugegriffen wird.

Als Ergänzung zum ursprünglichen MPEG-1-Verfahren gibt es mittlerweile MPEG-2 und MPEG-4, die sich aber an andere Zielgruppen richten. MPEG-1 wurde für den Videomarkt entwickelt, um Bildsequenzen mit mäßigem Detailgehalt und nicht zu gravierenden Szenenwechseln mit einer Datenrate von bis zu 1.5 MBit/s abdecken zu können; (z.B. Video von CD). Die Qualität der MPEG-1-Codierung einer Bildfolge entspricht in etwa der eines VHS-Recorders. Im Gegensatz hierzu ist MPEG-2 bei gleicher Vorgehensweise mit größeren Bildformaten und besserer Bildqualität bei einer Datenrate von bis zu 40 MBit/s auf Video-Übertragungen in Breitbandnetzen zugeschnitten. Die Forderung nach HDTV-Formaten (High Definition TV), die ursprünglich mit MPEG-3 abgedeckt werden sollte, wurde mittlerweile in MPEG-2 integriert. MPEG-4 ist ein skalierbares Format, das damit z.B. auch für die mobile Kommunikation in schmalbandigen Netzen geeignet ist. Das Spektrum der Skalierbarkeit reicht vom (künftigen) Handy bis HDTV, wobei die Datenrate und damit die Bildgröße und Güte des decodierten Videos den jeweiligen Gegebenheiten bzw. Erfordernissen angepasst wird. Dabei kann MPEG-4 im Vergleich zu den Vorgängerversionen noch höher komprimieren bzw. es liefert bei gleicher Bitrate eine bessere Bildqualität.

Weitere Aspekte von MPEG-4 sind der objektorientierte Ansatz mit Hilfe sogenannten ***audio-visueller Objekte*** (Avs) und die Möglichkeit, mit solchen Objekten zu interagieren und so z.B. Aktionen auszulösen.

7.2.10 Schlussbemerkungen

Neben den in diesem Kapitel aufgezählten Bilddatenformaten gibt es noch zahlreiche Alternativen, die aber zum Teil nur noch in geringem Umfange genutzt werden oder sich auf dem Markt (noch) nicht entsprechend durchsetzen konnten. So gibt es z.B. zur Wavelet-Kompression von Bildfolgen das kommerzielle und nicht standardisierte **Indeo**-Videokompressionsverfahren, das von Intel entwickelt wurde.

Im gegenseitigen Vergleich der unterschiedlichen Bilddatenformate kann hinsichtlich der Kompressionsrate generell gesagt werden, dass verlustbehaftete Verfahren deutlich günstiger abschneiden, als verlustfreie Verfahren. Bei realen Daten kann man mit den verlustfreien Verfahren selten um mehr als 20 Prozent komprimieren. In ungünstigen Fällen kann das codierte Bild sogar mehr Speicherplatz belegen, als das Original. Mit den verlustbehafteten Kompressionsverfahren sind (in Abhängigkeit vom Bildinhalt) häufig Kompressionen auf 10 bis 30 Prozent des ursprünglichen Platzbedarfs möglich. Der Benutzer muss bei der Einzelbildcodierung also in erster Linie entscheiden, ob er Qualitätsverluste hinnehmen kann und wie hoch diese sein dürfen.

Bei Bewegtbildern gibt es kaum eine Alternative zu MPEG, wobei für PC-Multimedia-Anwendungen nach wie vor das MPEG-1-Format ausreichend ist.