

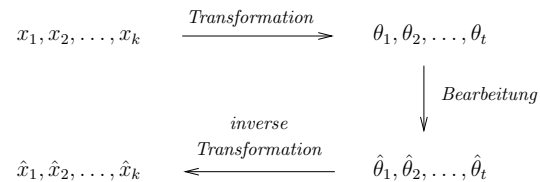
Algorithmische Verfahren zur Datenkompression

Dr. Maciej Liškiewicz
 Institut für Theoretische Informatik
 Universität zu Lübeck

Februar 2005

1 Transformcodierung und JPEG-Standard

Das allgemeine Konzept der *Transformation* oder *Umformung* ist sehr gut in der Mathematik und auch in anderen Gebieten bekannt. Das Schema für eine Transformation sieht allgemein folgendermaßen aus: Seien x_1, x_2, \dots, x_k Größen, wie z.B. Zahlen, Vektoren, Funktionen etc. Dann ändern wir diese Objekte mit Hilfe einer Transformation T und bekommen $\theta_1, \theta_2, \dots, \theta_t$, die möglicherweise ganz anders als p_i aussehen, die man aber einfach weiterverarbeiten kann. Schematisch kann man eine Transformation wie folgt darstellen:



Unser Ziel ist es, Transformationen zu finden, sodass man $\theta_1, \theta_2, \dots, \theta_t$ effizienter (verlustbehaftet) komprimieren kann als die Folge x_1, x_2, \dots, x_k direkt. Dann ist $\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_t$ eine Rekonstruktion des Komprimierungsverfahrens der Folge $\theta_1, \theta_2, \dots, \theta_t$, und $\hat{x}_1, \hat{x}_2, \dots, \hat{x}_k$ ist eine Rekonstruktion der Eingabe x_1, x_2, \dots, x_k . Viele der bislang vorgestellten Verfahren lassen sich auch als Transformationscodierungen begreifen. Eine konkrete verlustfreie Transformation haben wir auch schon explizit als solche kennen gelernt: die BWT.

Dennoch wollen wir das Grundprinzip anhand eines einfachen Beispiels erläutern, nämlich dem schon früher verwendeten, bei dem die Daten aus einer

Größe	Gewicht
163	85
188	94
150	75
175	85
140	65
200	102
170	80
125	55
100	40
125	77
173	74
155	70
190	82
160	60

Tabelle 1: Der Eingangs-Datensatz

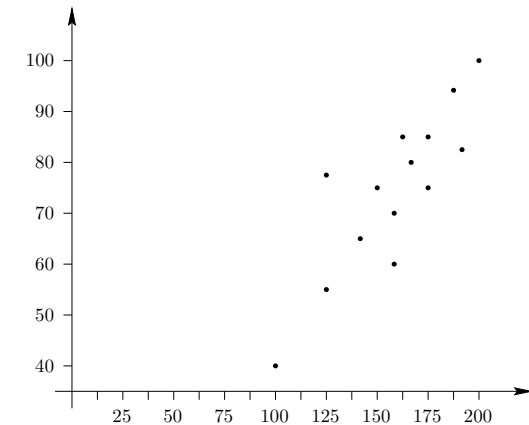


Abbildung 1: Graphische Darstellung der Eingangsdaten

x -Wert	y -Wert
184	3
210	0
168	0
195	-2
154	-4
225	2
188	-4
136	-7
107	-9
146	13
188	-11
170	-7
207	-12
170	-18

Abbildung 2: Die transformierten Werte

Folge von Paaren (Länge, Gewicht) von Personen bestehen. Ein möglicher Datensatz ist in Tabelle 1 aufgeführt. Betrachtet man Länge und Gewicht als (x, y) -Koordinaten, so sieht man in Bild 1, dass sich die Daten längs einer Geraden der Art $y = 0,5x$ häufen. Drehen wir den Datensatz also durch $\theta = \mathbf{A}\mathbf{x}$, wobei

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$$

der zweidimensionale Eingabevektor ist und

$$\mathbf{A} = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix}$$

die Drehmatrix sowie

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

der Vektor der Transformierten; speziell ist hier $\phi = \arctan(0,5) \approx 26,565^\circ$, d. h.

$$\mathbf{A} = \begin{bmatrix} 0,89442719 & 0,4472136 \\ -0,4472136 & 0,89442719 \end{bmatrix}$$

Auf diese Weise erhält man als Tabelle der transformierten Werte die Werte aus 2, was graphisch der in Bild 3 dargestellten Situation entspricht.

Wir beobachten bei den transformierten Werten, dass sich die „Energie“ in der ersten Komponente „ballt“, was intuitiv eine (verlustbehaftete) Komprimierung um den Faktor zwei gestattet. Täten wir dies tatsächlich und

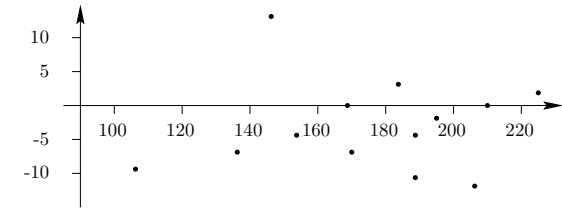


Abbildung 3: Die transformierten Werte

Größe	Gewicht
165	82
188	94
150	75
174	87
138	69
201	101
168	84
122	61
96	48
131	65
168	84
152	76
185	93
152	76

Tabelle 2: Die rücktransformierten Werte

wendeten auf die „komprimierte Folge“ die inverse Transformation, gegeben durch

$$\mathbf{A}^{-1} = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$$

an, so erhielten wir in etwa die Werte aus Tabelle 2.

In diesem Fall gilt:

$$\sum_{i=0}^{N-1} (x_i - \hat{x}_i)^2 = \sum_{i=0}^{N-1} (\theta_i - \hat{\theta}_i)^2$$

wobei

$$\hat{\theta}_i = \begin{cases} \theta_i & i = 0, 2, 4, \dots \\ 0 & \text{sonst} \end{cases}$$

und \hat{x}_i der x_i entsprechende rekonstruierte Wert ist. Diese Eigenschaft gilt allgemein für Transformationsmatrizen mit $A^{-1} = A^T$. Letztere Eigenschaft nennt man auch *Orthonormalität*.

1.1 Einfache Transformationen

Zuerst betrachten wir Transformationen für eindimensionale Folgen, wie sie zum Beispiel (digitalisierte) Sprache und allgemeiner Audio-Folgen. Es sei p_0, p_1, p_2, \dots die Eingabefolge liefern. Wir werden folgende *lineare Transformationen* betrachten: Für jeden Block

$$x_0 = p_\ell, x_1 = p_{\ell+1}, \dots, x_{N-1} = p_{\ell+N-1}$$

der Länge N , mit $\ell = 0, N, 2N, \dots$, transformieren wir x_0, x_1, \dots, x_{N-1} folgendermaßen:

$$\theta_n = \sum_{i=0}^{N-1} x_i a_{n,i},$$

wobei die Transformation vollständig durch die Konstanten $a_{n,i}$ definiert wird. Die Länge N hängt von praktischen Anwendungen ab. Es ist klar, dass mit größerem N auch die Transformation komplexer wird. Seien nun x und θ die Vektoren. Dann können wir eine lineare Transformation in Matrixform darstellen: $\theta = Ax$. Die inverse Transformation definiert eine Matrix B , so dass $x = B\theta$ gilt. Die Matrix B ist die Inverse zu A , das heißt $AB = BA = I$. Transformationscodierung ist eine der populärsten Methoden für die Komprimierung von Bildern. Deshalb werden wir bis Ende dieses Kapitels meistens den zweidimensionalen Fall der linearen Transformation betrachten.

Wir definieren solche Transformationen für einen gegebenen $N \times N$ Block

$$X = \begin{bmatrix} x_{0,0} & x_{0,1} & \dots & x_{0,N-1} \\ x_{1,0} & x_{1,1} & \dots & x_{1,N-1} \\ \vdots & & & \vdots \\ x_{N-1,0} & x_{N-1,1} & \dots & x_{N-1,N-1} \end{bmatrix}$$

folgendermaßen:

$$\Theta = AXA^T$$

und die inverse Transformation: $X = B\Theta B^T$. Alle Transformationen, die wir hier betrachten werden, sind orthonormal, das heißt, dass die inverse Matrix B (für den reellen Bereich) einfach die transponierte Matrix A^T ist:

$$B = A^{-1} = A^T.$$

Daher ist es einfach, die inverse Transformation zu bekommen:

$$X = A^T \Theta A$$

Die Zeilen der Transformationsmatrix nennen wir *Basisvektoren* und die Elemente nach der Transformation nennen wir oft *Transformationskoeffizienten*.

Man beachte, dass wir auf diese Art und Weise nur sog. *separable Transformationen* für den zweidimensionalen Fall erhalten; denn die allgemeine Transformationsformel lautet

$$\Theta_{k,\ell} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} X_{i,j} a_{i,j,k,\ell},$$

(dies wäre ein Tensorprodukt) und wir betrachten als Spezialisierung:

$$\Theta_{k,\ell} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} X_{i,j} a_{k,i} a_{\ell,j}.$$

Die Wirksamkeit einer Transformation hängt davon ab,

- wie groß die *Dekorrelation*, also die Reduktion der Korrelation, der Eingabepixel (oder Signale) ist und
- wieviel *Energiebündelung* (engl.: energy compaction) die Transformation ergibt.

Die Korrelation in einer Eingabefolge ist die Quelle der Redundanz, die in einem effizienten Komprimierungsverfahren entfernt werden soll.

Eine wichtige Eigenschaft der orthonormalen Transformationen ist, dass solche Transformationen die Energie erhalten:

$$\sum x_{i,j}^2 = \sum \theta_{i,j}^2.$$

Die Transformationen sollen nur die Energie „nach links oben“ in der Matrix verschieben, also dort bündeln.

Beispiel 1.1 Betrachten wir die folgende Transformationsmatrix:

$$A = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

A ist orthonormal, weil

$$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^T = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^T \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Die erste Zeile der Matrix $(1/\sqrt{2}, 1/\sqrt{2})$ entspricht einem Tiefpass und die zweite Zeile $(1/\sqrt{2}, -1/\sqrt{2})$ entspricht einem Hochpass. (Beide Begriffe werden wir weiter unten genauer besprechen.) Betrachten wir eine Folge, in der beide Elemente gleich sind. Nach der Transformation soll dann das zweite Element Null sein, d. h., die Energie ist vollständig in der ersten Komponente gebündelt. Es sei (α, α) eine solche Eingabefolge. Es gilt:

$$\theta = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \alpha \\ \alpha \end{bmatrix} = \begin{bmatrix} \sqrt{2}\alpha \\ 0 \end{bmatrix}$$

Wir diskutieren jetzt den zweidimensionalen Fall.

Beispiel 1.2 Wir nehmen die selbe Transformationsmatrix wie im vorigen Beispiel. Für die Eingabematrix $\begin{bmatrix} \alpha & \alpha \\ \alpha & \alpha \end{bmatrix}$ erhalten wir die folgende transformierte:

$$\Theta = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \alpha & \alpha \\ \alpha & \alpha \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^T = \begin{bmatrix} 2\alpha & 0 \\ 0 & 0 \end{bmatrix}$$

Eine interessante Eigenschaft ist, dass nach der Transformation der oberen Eingaben sich die ganze Energie im linken oberen Eck der Matrix konzentriert.

Aus historischen Gründen heißt der $\theta_{0,0}$ -Koeffizient (im linken oberen Eck der Matrix der Transformierten Werte) *DC-Koeffizient* oder *Niederfrequenz-Koeffizient* und die anderen *AC-Koeffizienten* oder *Hochfrequenz-Koeffizienten*. DC steht für *direct current*, zu deutsch Gleichstrom, und AC steht für *alternating current* (Wechselstrom). Einen Grund für diese Namensgebung sieht man, wenn man eine inverse Transformation für A betrachtet:

$$X = A\Theta A^T = \frac{1}{2} \begin{bmatrix} \theta_{0,0} + \theta_{0,1} + \theta_{1,0} + \theta_{1,1} & \theta_{0,0} - \theta_{0,1} + \theta_{1,0} - \theta_{1,1} \\ \theta_{0,0} + \theta_{0,1} - \theta_{1,0} - \theta_{1,1} & \theta_{0,0} - \theta_{0,1} - \theta_{1,0} + \theta_{1,1} \end{bmatrix}$$

Das heißt:

$$X = \theta_{0,0}\alpha_{0,0} + \theta_{0,1}\alpha_{0,1} + \theta_{1,0}\alpha_{1,0} + \theta_{1,1}\alpha_{1,1},$$

wobei

$$\alpha_{0,0} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad \alpha_{0,1} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

$$\alpha_{1,0} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \quad \alpha_{1,1} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

Beobachten Sie, dass alle Elemente der $\alpha_{0,0}$ Matrix gleich sind. Daher kommt die DC-Benennung.

1.2 Spezielle Transformationen für Bildverarbeitung

Wir stellen die wichtigsten Transformationen vor, die man für die Bildverarbeitung benutzt. Weitere Einzelheiten zu diesem Thema finden Sie z. B. in [3].

1.2.1 Karhunen-Loève-Transformation KLT

KLT ist nachweislich optimal hinsichtlich zweier Kriterien:

- Sie dekorreliert die Daten am besten und
- sie bündelt damit am besten die Information in den niederfrequenten Bereich.

Die entsprechenden mathematischen Messgrößen η_C und η_E werden wir weiter unten einführen.

Die wichtigsten Anfangsschritte der Transformation sind:

1. Berechne Kovarianzmatrix $COV(X)$ der Daten.

2. Bestimme die Eigenvektoren.

Das alles gilt jeweils für jedes Bild oder für jede Bildgruppe, da die Basisvektoren nur für die jeweilige Datenmenge charakteristisch sind. Zusätzlich müssen noch die Basisvektoren zusammen mit den Kompressionseingaben als Begleitinformationen verschickt werden. Deshalb ist diese Methode in der Praxis nicht nützlich, stellt aber doch eine wichtige Messlatte für die Güte von Kompressionsverfahren dar.

Wir geben jetzt die wichtigsten Begriffe für eine Bildtransformation, und dann zeigen wir, wie die Karhunen-Loève-Transformation aussieht.

Wir erinnern zunächst an die Begriffe *Mittelwert* \bar{x} und *Varianz* σ^2 für das ganze Bild ($N \times M$ Pixel):

$$\bar{x} = \frac{1}{N \times M} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} x_{i,j}$$

$$\sigma^2 = \frac{1}{N \times M} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (x_{i,j} - \bar{x})^2.$$

Jetzt betrachten wir –als Vorbereitung für den zweidimensionalen Fall– die Korrelation im Eindimensionalen.

Für eine (eindimensionale) Folge x ist die Varianz:

$$\sigma^2 = \sigma_{xx}^2 = E[(x - \bar{x})^2].$$

Für zwei Folgen x und y definieren wir als ihre *Kovarianz*

$$\sigma_{xy}^2 = E[(x - \bar{x})(y - \bar{y})].$$

Hier und im Folgenden werden wir als „Erwartungswert“ stets den Mittelwert ansetzen, was ja auch ein üblicher Schätzer ist. Ähnlich „lässig“ verfahren wir mit anderen Begriffen aus Wahrscheinlichkeitstheorie und Statistik.

Ist $\sigma_{xy}^2 = 0$, so sind die beiden Folgen *unkorreliert*. Jetzt betrachten wir als Beispiel drei Folgen x, y und z . Dann erhalten wir die folgende *Kovarianzmatrix*:

$$COV(X) = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{xz}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{yz}^2 \\ \sigma_{zx}^2 & \sigma_{zy}^2 & \sigma_{zz}^2 \end{bmatrix}$$

Für k Folgen ist $COV(X)$ analog eine $k \times k$ -Matrix.

Jetzt definieren wir für die Folge

$$x_1, x_2, \dots, x_M$$

als *k-te Kovarianz*

$$\sigma_{1,k+1}^2 = \sigma_{k+1,1}^2 = \frac{1}{M-k} \sum_{i=1}^{M-k} (x_i - \bar{x})(x_{i+k} - \bar{x}),$$

d. i. die Kovarianz für die Folgen x_1, x_2, \dots, x_{M-k} und $x_{k+1}, x_{k+2}, \dots, x_M$, wobei wir als Mittelwert für beide Folgen (vereinfachend) \bar{x} nehmen. Vereinfachend setzen wir weiter:

$$\sigma_{1,k+1}^2 = \sigma_{k+1,1}^2 = \frac{1}{M} \sum_{i=1}^{M-k} (x_i - \bar{x})(x_{i+k} - \bar{x}).$$

Für $k \ll M$ ist das immer eine gute Abschätzung. Dann bekommen wir:

$$\sigma_{11}^2 = \sigma_{22}^2 = \dots = \sigma_{kk}^2 = \sigma^2$$

$$\sigma_{12}^2 = \sigma_{23}^2 = \dots = \sigma_{k-1,k}^2 = \sigma_1^2$$

usw., wobei wir $\sigma_{\ell,k}^2$ ähnlich definieren wie $\sigma_{1,k+1}^2$ nur für die Folgen, die mit x_ℓ bzw. $x_{\ell+k}$ starten. Dann ist die *Kovarianzmatrix einer Folge*:

$$COV(X) = \begin{bmatrix} \sigma^2 & \sigma_1^2 & \sigma_2^2 & \dots & \sigma_{k-1}^2 \\ \sigma_1^2 & \sigma^2 & \sigma_1^2 & \dots & \sigma_{k-2}^2 \\ \vdots & \vdots & \vdots & & \vdots \\ \sigma_{k-1}^2 & \sigma_{k-2}^2 & \sigma_{k-3}^2 & \dots & \sigma^2 \end{bmatrix}$$

Wir normalisieren die Matrix:

$$COV(X) = \sigma^2 \begin{bmatrix} 1 & \rho_1 & \rho_2 & \dots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \dots & \rho_{k-2} \\ \vdots & \vdots & \vdots & & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \dots & 1 \end{bmatrix},$$

wobei $\rho_k = \sigma_k^2 / \sigma^2$ der *k-te Korrelationskoeffizient* ist. Zur Bildverarbeitung passt der folgende Fall erfahrungsgemäß sehr gut:¹

$$\rho_k = \rho^k,$$

¹Genau genommen wird dabei mit der Annahme gearbeitet, die Eingabefolge sei ein stationärer Markov-Prozess k -ter Ordnung. In der Praxis bewährt sich diese an und für sich falsche Annahme.

mit $\rho = \rho_1$. Wir nennen ρ *Zwischen-Element-Korrelation*. Dann definieren wir die *Korrelationsmatrix*

$$COR(X) = \begin{bmatrix} 1 & \rho & \rho^2 & \dots & \rho^{k-1} \\ \rho & 1 & \rho & \dots & \rho^{k-2} \\ \vdots & \vdots & \vdots & & \vdots \\ \rho^{k-1} & \rho^{k-2} & \rho^{k-3} & \dots & 1 \end{bmatrix}$$

Für den zweidimensionalen Fall verallgemeinern sich diese Begriffe mit Hilfe von vertikalen und horizontalen Folgen. Die Verallgemeinerung ist nicht trivial und wir werden sie hier nicht vertiefen.

Daher betrachten wir jetzt nur (noch) den eindimensionalen Fall.

Es seien $COV(X) = (X_{i,j})$ und $COV(Y) = (Y_{i,j})$ die Kovarianz-Matrizen für eine eindimensionale Folge und für ihre Transformation. Dann setzen wir:

$$\Sigma X = \sum_{\substack{1 \leq i, j \leq N \\ i \neq j}} |X_{i,j}|$$

und

$$\Sigma Y = \sum_{\substack{1 \leq i, j \leq N \\ i \neq j}} |Y_{i,j}|$$

und definieren damit die *Dekorrelationswirkung* η_C folgendermaßen:

$$\eta_C = 1 - \frac{\Sigma Y}{\Sigma X}.$$

Eine andere Größe, die die Qualität einer Transformation beschreibt, ist der *Energie-Bündelungs-Koeffizient* in den ersten M von N diagonalen Komponenten:

$$\eta_E = \frac{\sum_{j=1, k=j}^M Y_{j,k}}{\sum_{j=1, k=j}^N Y_{j,k}}.$$

Die Zeilen der Karhunen-Loève-Transformation sind die Eigenvektoren der *Autokorrelationsmatrix*. Für jedes $i = 0, 1, \dots, N-1$ lösen wir folgende Gleichung

$$\tan N\omega_i = \frac{-(1 - \rho^2) \sin \omega_i}{(1 + \rho^2) \cos \omega_i - 2\rho},$$

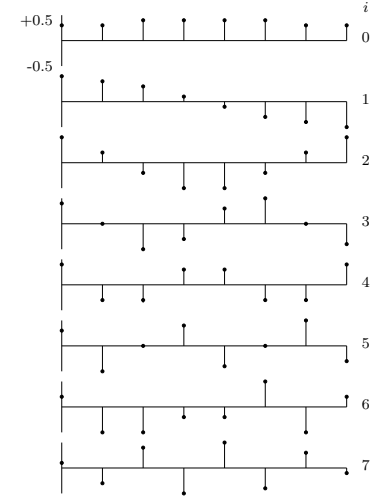


Abbildung 4: Die acht Basisvektoren der KLT für $\rho = 0,91$

wobei ρ die Zwischen-Element-Korrelation ist, und dann benutzen wir die Lösungen, um die Eigenvektoren zu berechnen. Dafür setzen wir:

$$\lambda_i = \frac{1 - \rho^2}{1 + \rho^2 - 2\rho \cos \omega_i}.$$

Basisvektoren für $0 \leq i, j \leq N-1$ sind dann:

$$a_{i,j} = \left(\frac{2}{N + \lambda_i} \right)^{1/2} \sin \left[\omega_i \left(j - \frac{N-1}{2} \right) + \frac{(i+1)\pi}{2} \right]$$

In Bild 4 sehen Sie die Werte der Basisvektoren für $N = 8$ und $\rho = 0,91$.

Intuitiv stellt die Autokorrelationsmatrix gerade die Abhängigkeiten zwischen aufeinander folgenden Gliedern einer Folge dar. Da die Eigenvektoren (als neue Basisvektoren) immer die „Richtungen“ angeben, in die Folge „strebt“, leistet KLT die gewünschte (optimale) Energiebündelung.

1.2.2 Diskrete Fouriertransformation

Diese ist die einzige komplexzahlige Transformation, die wir behandeln werden. Die (komplexe) Transformationsmatrix lautet:

$$a_{\ell,k} = \sqrt{1/N} \left(\cos \frac{2\pi\ell k}{N} - i \sin \frac{2\pi\ell k}{N} \right).$$

Die Basisvektoren für $N = 8$ sehen folgendermaßen aus: Die reelle Basis:

$$\begin{bmatrix} 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 \\ 0,35 & 0,25 & 0,00 & -0,25 & -0,35 & -0,25 & 0,00 & 0,25 \\ 0,35 & 0,00 & -0,35 & 0,00 & 0,35 & 0,00 & -0,35 & 0,00 \\ 0,35 & -0,25 & 0,00 & 0,25 & -0,35 & 0,25 & 0,00 & -0,25 \\ 0,35 & -0,35 & 0,35 & -0,35 & 0,35 & -0,35 & 0,35 & -0,35 \\ 0,35 & -0,25 & 0,00 & 0,25 & -0,35 & 0,25 & 0,00 & -0,25 \\ 0,35 & 0,00 & -0,35 & 0,00 & 0,35 & 0,00 & -0,35 & 0,00 \\ 0,35 & 0,25 & 0,00 & -0,25 & -0,35 & -0,25 & 0,00 & 0,25 \end{bmatrix}$$

und die imaginäre Basis

$$\begin{bmatrix} 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 \\ 0,00 & 0,25 & 0,35 & 0,25 & 0,00 & -0,25 & -0,35 & -0,25 \\ 0,00 & 0,35 & 0,00 & -0,35 & 0,00 & 0,35 & 0,00 & -0,35 \\ 0,00 & 0,25 & -0,35 & 0,25 & 0,00 & -0,25 & 0,35 & -0,25 \\ 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 & 0,00 \\ 0,00 & -0,25 & 0,35 & -0,25 & 0,00 & 0,25 & -0,35 & 0,25 \\ 0,00 & -0,35 & 0,00 & 0,35 & 0,00 & -0,35 & 0,00 & 0,35 \\ 0,00 & -0,25 & -0,35 & -0,25 & 0,00 & 0,25 & 0,35 & 0,25 \end{bmatrix}$$

Es scheint so zu sein, dass man den imaginären Anteil nicht einfach vernachlässigen kann, so wie dies in der Praxis wohl manchmal passiert. Das Problem der „Datenaufblähung“ durch Einführung eines Imaginärteils umgeht die diskrete Cosinustransformation:

1.2.3 DCT — Diskrete Cosinus-Transformation

Die (reelle) Transformationsmatrix lautet hier:

$$a_{i,j} = \begin{cases} \sqrt{1/N} & i = 0, j = 0, 1, \dots, N-1 \\ \sqrt{2/N} \cos \frac{(2j+1)i\pi}{2N} & i = 1, \dots, N-1, j = 0, 1, \dots, N-1. \end{cases}$$

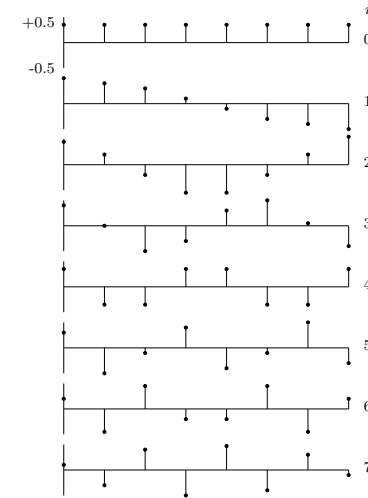


Abbildung 5: Die acht Basisvektoren der DCT

Für den 8×8 -Fall ergeben sich als Basisvektoren:

$$A = \begin{bmatrix} 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 & 0,35 \\ 0,49 & 0,42 & 0,28 & 0,10 & -0,10 & -0,28 & -0,42 & -0,49 \\ 0,46 & 0,19 & -0,19 & -0,46 & -0,46 & -0,19 & 0,19 & 0,46 \\ 0,42 & -0,10 & -0,49 & -0,28 & 0,28 & 0,49 & 0,10 & -0,42 \\ 0,35 & -0,35 & -0,35 & 0,35 & 0,35 & -0,35 & -0,35 & 0,35 \\ 0,28 & -0,49 & 0,10 & 0,42 & -0,42 & -0,10 & 0,49 & -0,28 \\ 0,19 & -0,46 & 0,46 & -0,19 & -0,19 & 0,46 & -0,46 & 0,19 \\ 0,10 & -0,28 & 0,42 & -0,49 & 0,49 & -0,42 & 0,28 & -0,10 \end{bmatrix}$$

Ihre graphische Darstellung entnehmen Sie bitte Bild 5.

Vergleichen Sie die Werte der Basisvektoren der diskreten Cosinus-Transformation mit den Werten für KLT in Bild 4. Sie werden bemerken, dass sich die Werte von DCT und KLT kaum unterscheiden, was die gute Qualität der DCT erklärt.

Beispiel 1.3 Wir betrachten jetzt drei Beispiele zur DCT-Transformation:

1. Zwei aneinanderstoßende Flächen

$$X = \begin{bmatrix} 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \\ 10 & 10 & 10 & 10 & 128 & 128 & 128 & 128 \end{bmatrix}$$

besitzen folgende Cosinus-Transformierte:

$$\Theta = \begin{bmatrix} 552,0 & -427,7 & 0,0 & 150,2 & 0,0 & -100,4 & 0,0 & 85,1 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \end{bmatrix}$$

2. Ein Block des Bildes „Brücke“

$$X = \begin{bmatrix} 94 & 96 & 113 & 135 & 196 & 116 & 110 & 106 \\ 91 & 125 & 120 & 153 & 192 & 135 & 108 & 124 \\ 87 & 119 & 144 & 122 & 190 & 131 & 115 & 132 \\ 88 & 93 & 141 & 100 & 168 & 149 & 128 & 122 \\ 95 & 93 & 139 & 169 & 172 & 147 & 154 & 135 \\ 87 & 100 & 124 & 173 & 180 & 132 & 173 & 178 \\ 104 & 85 & 112 & 120 & 167 & 158 & 132 & 166 \\ 112 & 83 & 124 & 123 & 154 & 152 & 148 & 165 \end{bmatrix}$$

wird transformiert in:

$$\Theta = \begin{bmatrix} 1049,9 & -125,7 & -121,2 & 4,7 & 50,1 & -40,0 & 2,4 & 55,2 \\ -30,4 & 59,0 & -48,7 & 6,0 & -0,2 & -35,8 & -22,7 & 13,3 \\ -17,2 & 5,7 & 12,8 & 16,4 & 24,6 & -11,0 & 0,6 & -5,4 \\ 15,7 & -14,8 & -17,5 & -9,4 & 25,7 & 26,1 & -30,5 & -10,9 \\ -19,9 & 5,6 & -0,6 & 14,8 & -13,1 & 22,2 & 14,5 & 12,3 \\ -30,6 & -3,6 & 13,5 & 17,2 & -8,3 & -18,8 & 20,3 & 19,7 \\ 11,1 & -6,8 & 2,1 & -19,6 & 4,3 & 5,6 & -16,0 & 13,4 \\ 0,5 & 9,5 & -7,4 & 2,8 & 4,0 & 1,0 & 5,5 & -0,2 \end{bmatrix}$$

3. Ein einzelner Punkt

$$X = \begin{bmatrix} 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 128 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 & 10 & 10 & 10 \end{bmatrix}$$

ergibt:

$$\Theta = \begin{bmatrix} 94,8 & 4,1 & -19,3 & -11,6 & 14,8 & 17,3 & -8,0 & -20,5 \\ 4,1 & 1,1 & -5,3 & -3,2 & 4,1 & 4,8 & -2,2 & -5,6 \\ -19,3 & -5,3 & 25,2 & 15,1 & -19,3 & -22,7 & 10,4 & 26,7 \\ -11,6 & -3,2 & 15,1 & 9,1 & -11,6 & -13,6 & 6,3 & 16,1 \\ 14,8 & 4,1 & -19,3 & -11,6 & 14,8 & 17,3 & -8,0 & -20,5 \\ 17,3 & 4,8 & -22,7 & -13,6 & 17,3 & 20,4 & -9,4 & -24,1 \\ -8,0 & -2,2 & 10,4 & 6,3 & -8,0 & -9,4 & 4,3 & 11,1 \\ -20,5 & -5,6 & 26,7 & 16,1 & -20,5 & -24,1 & 11,1 & 28,4 \end{bmatrix}$$

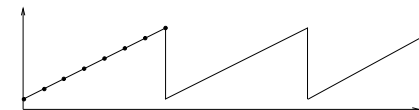
Diskrete Cosinus-Transformation und diskrete Fourier-Transformation hängen eng zusammen: Spiegelt man die N -Punkt-Folge einer DFT am rechten Rand, so erhält man die „zugehörige“ $2N$ -Punkt-Folge der DCT.

Bemerkung 1.4 Die wichtigsten Unterschiede zwischen DCT und DFT sind folgende:

1. DFT generiert komplexe Zahlen, während DCT nur reelle Zahlen ausgibt.
2. DFT nimmt an, dass die Funktion, die aus x_0, x_1, \dots, x_{N-1} rekonstruiert wird, periodisch ist (mit Periode N). So betrachtet DFT die Folge

$$8, 16, 24, 32, 40, 48, 56, 64$$

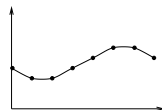
als die Werte der folgenden periodischen Funktion:



8	7	5	3	1	1	0	0
7	5	3	2	1	0	0	0
4	3	2	1	1	0	0	0
3	3	2	1	1	0	0	0
2	1	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Tabelle 3: Typische Bitverteilung für eine 8×8 -Transformation

Die inverse Transformation für die „bearbeiteten Werte“ des Vektors Ax (z. B. nach Quantisierung) gibt daher die Werte der folgenden Funktion:



DCT hat solche Eigenschaft nicht. Wir betrachten DCT als die ersten N reellen Werte der DFT für die Folge:

$$x_0, x_1, \dots, x_{N-1}, x_{N-1}, \dots, x_1, x_0$$

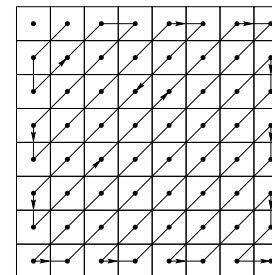
und deshalb entfernen wir diese Nichtstetigkeiten.

3. Die Komplexität der DCT ist höher als die der DFT.

Bemerkung 1.5 In der Praxis benutzt Software zur Berechnung der DCT Festpunktarithmetik. Der „Weltrekord“ für die schnellste Ausführung der DCT liegt bei 11 Multiplikationen und 29 Additionen [C. Loeffler, A. Ligtenberg and G. Moschytz, *Practical Fast 1-D DCT Algorithms with 11 Multiplications*, Proc. Int'l. Conf. on Acoustics, Speech, and Signal Processing 1989 (ICASSP '89), pp. 988-991].

1.3 Bit-Verteilung

Wie aus den Beispielen ersichtlich, ist der Informationsgehalt der Einträge der transformierten Matrix unterschiedlich; ähnlich wie bei der zuvor betrachteten Teilbandkompression massiert sich die Energie im linken oberen Eck, und das ist ja auch eines der Kriterien für eine gute Transformation. Das

Abbildung 6: Zickzack-Abtastung eines 8×8 -Musters

bedeutet wiederum, dass es sinnvoll ist, bei der quantisierten Übertragung dieser transformierten Koeffizienten unterschiedlich viele Bits zu verwenden. Diese Beobachtung führt zur Anwendung des Bitverteilungsalgorithmus aus der Vorlesung über Teilbandcodierung und wird in diesem Kontext *Zonenabtastung* (engl.: zonal sampling) genannt. Tabelle 3 zeigt solch eine typische Bitverteilung.

Chen und Pratt haben ein anderes Bitverteilungsverfahren vorgeschlagen, bei dem die zu übertragenden Werte des transformierten Bildes zunächst quantisiert werden (mit einem Quantisierer, der u. a. Null als Repräsentanten eines um den Nullpunkt symmetrischen Intervalls hat) und dann in der in Bild 6 angegebenen Weise zickzackartig abgetastet werden. Man kann annehmen, dass die sich so ergebene abgetastete Zahlenfolge ab einem gewissen Index konstant Null ist (dieser Annahme liegt ja offenkundig die in Tab. 3 angegebene fixe Bitverteilung zugrunde); daher erscheint es sinnvoll, eine solche konstante Nullfolge durch ein Sonderzeichen, hier EOB (*Blockende*, engl.: end of block) genannt, dem Empfänger anzuzeigen.

Der Vorteil dieses auch in JPEG verwendeten Schemas liegt darin, dass evtl. noch vorhandene „hochfrequente“ Informationen nicht automatisch vernachlässigt werden.

1.4 JPEG

Wir haben jetzt das nötige Rüstzeug zusammen, um den von der JPEG vorgeschlagenen Standard (im Folgenden auch JPEG genannt) verstehen zu können.

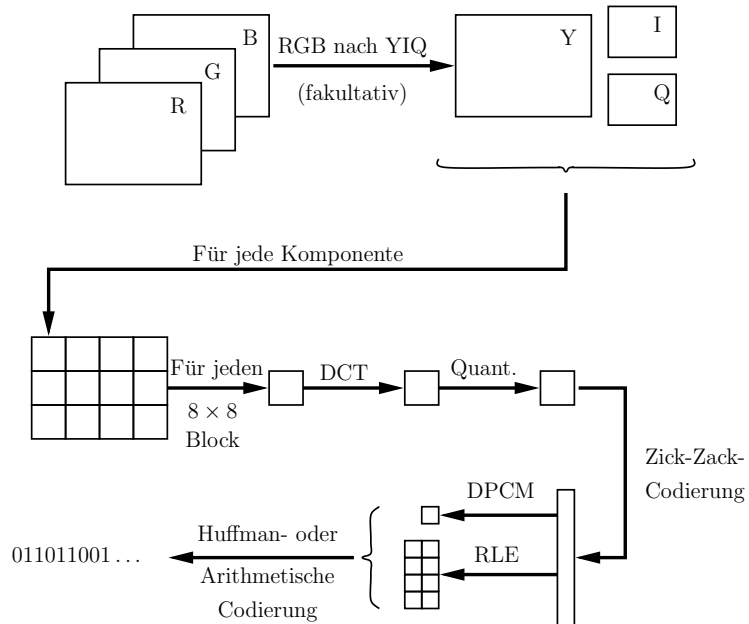


Abbildung 7: Das Schema von JPEG

124	125	122	120	122	119	117	118
121	121	120	119	119	120	120	118
126	124	123	122	121	121	120	120
124	124	125	125	126	125	124	124
127	127	128	129	130	128	127	125
143	142	143	142	140	139	139	139
150	148	152	152	152	152	150	151
156	159	158	155	158	158	157	156

Tabelle 4: Ein 8 × 8-Block aus dem Sena-Bild...

Die wichtigsten Schritte der JPEG (schematisch dargestellt in Bild 7) sind folgende:

- diskrete Cosinus-Transformation,
- Quantisierung,
- Differentialcodierung auf DC-Komponenten,
- Zickzack-Abtastung der AC-Komponenten,
- Lauflängencodierung auf AC-Komponenten sowie
- Huffman-Codierung oder arithmetische Codierung.

Schauen wir uns die einzelnen Schritte des Schemas 7 genauer an:

1. Der erste Schritt ist fakultativ: Eine Umwandlung von RGB nach YIQ. RGB und YIQ sind dreidimensionale Räume, die die *Farben* und die *Helligkeit* beschreiben. RGB hat ein passendes Format für die Hardware, wohingegen YIQ eher der menschlichen Wahrnehmung entspricht. I stellt einen Farbausgleich zwischen Orange und Cyan dar. Die Q-Komponente beschreibt einen Ausgleich zwischen Grün und Magenta. I und Q zusammen stellen die Farbe jedes Pixels dar. Die Y-Komponente beschreibt die Helligkeit des Pixels.
Um eine Umwandlung von RGB nach YIQ zu erhalten, muss man entscheiden, wieviele Bits den einzelnen Komponenten zukommen. Im allgemeinen Fall gewähren wir viermal soviel Bits für die Y- wie für die I- oder die Q-Komponente; denn die Augen sind empfindlicher für Änderungen der Helligkeit als für Änderungen der Farbe.
2. Teile das Bild in Pixelblöcke der Größe 8 × 8.

39,88	6,56	-2,24	1,22	-0,37	-1,08	0,79	1,13
-102,43	4,56	2,26	1,12	0,35	-0,63	-1,05	-0,48
37,77	1,31	1,77	0,25	-1,50	-2,21	-0,10	0,23
-5,67	2,24	-1,32	-0,81	1,41	0,22	-0,13	0,17
-3,37	-0,74	-1,75	0,77	-0,62	-2,65	-1,30	0,76
5,98	-0,31	-0,45	-0,77	1,99	-0,26	1,46	0,00
3,97	5,52	2,39	-0,55	-0,051	-0,84	-0,52	-0,13
-3,43	0,51	-1,07	0,87	0,96	0,09	0,33	0,01

Tabelle 5: liefert diese DCT-Koeffizienten nach JPEG.

16	11	10	16	24	40	51	61
12	12	14	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
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Tabelle 6: Die Standard-Quantisiermatrix von JPEG für Helligkeit

3. Führe DCT für jeden Block aus.

(Dabei wird zuvörderst jeder Pixelwert von 0 bis $2^P - 1$ durch Subtraktion von 2^{P-1} in einen um den Nullpunkt symmetrischen Bereich abgebildet.) Man beobachte das Beispiel eines Blocks aus dem Sena-Bild in den Tabellen 4 und 5.

Warum benutzt JPEG DCT und nicht DFT?

Zur Beantwortung dieser Frage diskutieren wir die Punkte von Bemerkung 1.4. Natürlich ist Punkt (a) kein Problem und die entscheidenden Unterschiede sind (b) und (c). Das JPEG-Komitee hat DCT gewählt wegen (b) und trotz (c). Man kann empirisch zeigen, dass die Pixel eines Bildes nicht einer periodischen Funktion entsprechen und daher der Grundannahme der DFT widersprechen.

4. Quantisierung

Der Quantisierfehler ist die Hauptquelle der Verzerrung in JPEG. Das Verfahren benutzt eine 8×8 Quantisiermatrix Q und das heißt, dass JPEG für jeden 8×8 -Block X die Werte

$$\ell_{i,j} = \lfloor \Theta_{i,j}/Q_{i,j} + 0,5 \rfloor$$

2	1	0	0	0	0	0	0
-9	0	0	0	0	0	0	0
3	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	0

Tabelle 7: Die Matrix $\ell_{i,j}$ der Quantisierlabels des Sena-Blocks...

32	11	0	0	0	0	0	0
-108	0	0	0	0	0	0	0
42	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	0

Tabelle 8: ...und die zugehörigen Repräsentanten

berechnet, wobei Θ die Matrix X nach der diskreten Cosinus-Transformation ist. Die übliche JPEG-Quantisiermatrix Q können Sie Tabelle 6 entnehmen.

Im Beispiel in Tabelle 5 ist $\Theta_{0,0} = 39,88$, d.h.

$$\ell_{0,0} = \left\lceil \frac{39,88}{16} + 0,5 \right\rceil = \lceil 2,9925 \rceil = 2.$$

Der zugehörige Repräsentant ist daher $32 = 16 * 2$, und der Quantisierfehler wäre hier 7,88. Die weiteren Daten sind in Tabelle 7 abgelegt; die Matrix der zugehörigen Repräsentanten ist in Tabelle 8 aufgeführt.

JPEG definiert zwei Default-Quantisiermatrizen: eine für die Helligkeit (diese sehen Sie in Tab. 6), das heißt die Y-Komponente, und eine für die Farben, das heißt für die I- und Q-Komponenten.

Es ist möglich, in JPEG eine eigene Quantisiermatrix definieren. Dann fügt man diese Matrix in den Kopf des codierten Bildes.

Unten sehen Sie die Werte nach der Quantisierung für den Block „Brücke“ (vgl. Beispiel 1.3) und die Default-Matrix Q :

66	-11	-12	0	2	-1	0	1
-3	5	-3	0	0	-1	0	0
-1	0	1	1	1	0	0	0
1	-1	-1	0	1	0	0	0
-1	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

5. DPCM auf DC-Komponenten

DC-Komponenten sind groß und haben unterschiedliche Werte. Man kann aber sehen, dass viele DC-Komponenten ähnliche Werte wie ihre Vorgänger haben. Deshalb wählt man anstelle der direkten Codierung der DC-Komponenten ein DPCM-Verfahren: JPEG codiert die Differenzen zwischen Nachbarkomponenten. Diese Differenzen haben die Tendenz, klein zu sein.

Die Differenzen komprimiert man mit Hilfe des Huffman-Verfahrens oder der arithmetischen Codierung.

6. Zickzack-Abtastung der AC-Komponenten

Hierdurch wird der zweidimensionale Block in einen eindimensionalen Vektor umgewandelt, so dass diese Konversion die Energie des Blockes im Anfang des Vektors bündelt.

7. Lauflängencodierung auf AC-Komponenten

Man kann sehen, dass viele AC-Komponenten Null sind. JPEG codiert diese Werte als Paare

$$(skip, value),$$

123	122	122	121	120	120	119	119
121	121	121	120	119	118	118	118
121	121	120	119	119	118	117	117
124	124	123	122	122	121	120	120
130	130	129	129	128	128	128	127
141	141	140	140	139	138	138	137
152	152	151	151	150	149	149	148
159	159	158	157	157	156	155	155

Tabelle 9: Rekonstruktion des Sena-Blocks

wobei *skip* die Anzahl der Nullen und *value* die nächste Nicht-Null-Komponente ist. (0, 0) steht für das Ende. Dann komprimiert JPEG diese Folgen mit Hilfe des Huffman-Verfahrens oder der arithmetischen Codierung.

Die rekonstruierten Werte für den Sena-Block finden Sie in Tabelle 9.

Es sei abschließend erwähnt, dass es auch eine JPEG-Variante gibt, die die fortschreitende Bildübertragung unterstützt, s. [5].

1.5 Ein Vergleich von Verfahren zur Bildkompression

„Ein Bild sagt mehr als tausend Worte“ beschreibt treffend den Wert visueller Information für den Menschen. Leider umfasst ein hochaufgelöstes Digitalbild auch weit mehr Datenbits als nämliche tausend Worte, was die Notwendigkeit des Einsatzes von Kompressionsverfahren unterstreicht. Manche Anwendungen, z. B. im medizinischen oder militärischen Bereich erfordern eine getreue Wiedergabe des (digitalisierten) Originalbildes, da verlustbehaftete Kompressionsverfahren dort nicht tolerierbare Störungen mit sich bringen („Ist jener Fleck dort ein kleines Geschwür oder ein *Artefakt*, also eine Auswirkung der Fehlerbehaftung des Kompressionsverfahrens?“) In WWW-Anwendungen sind typischerweise gewisse Verzerrungen tolerierbar (z. B. in elektronischen Zeitungen); als Kompromiss bietet sich die fortschreitende Bildübertragung an.

Prädikative Methoden (wie Differentialcodierung oder verlustfreier JPEG-Standard), bei denen im wesentlichen codierte Differenzen des aktuellen Pixels vom „erwarteten Pixel“ (das aus der Kenntnis benachbarter Pixel errechnet wird) übertragen werden, sind im wesentlichen in Anwendungen im Einsatz, wo nur geringe Störungen toleriert werden können und dafür geringe Kompressionsfaktoren annehmbar sind.

Für mittlere Kompressionsraten sind Umformungs- und Teilbandcodierungen sehr geeignet. Umformungsbasierte Kompressionsverfahren wie JPEG arbeiten auf Datenblöcken und zeigen daher bei hohen Kompressionsraten Artefakte an den Blockrändern (*Blockeffekt*, engl.: blocking effect), welche Multiresolutionsansätze nicht aufweisen. Besonders deutlich erscheint der Blockeffekt in „ruhigen“ Bereichen. Andererseits neigen Teilbandcodierungen dazu, aufgrund der nicht „perfekten“ Filter *Welleneffekte* (engl.: ringing effect) an Rändern aufzuweisen. Durch geeignete Abstimmung der Filter aufeinander lässt sich dieser Effekt jedoch minimieren. Durch Techniken wie bei EZW lässt sich eine fortschreitende Bildübertragung erreichen.

Fraktale Kompressionsverfahren zeigen gute Leistungen bei relativ hohen Kompressionsraten (70 bis 80), die Bilder erscheinen dann aber oft „verschmiert“ im Vergleich zum Original. Des weiteren erfordern sie einen hohen Rechenaufwand auf Seiten des Codierers und sind daher eher für *asymmetrische Anwendungen* wie im Multimediabereich geeignet, bei denen typischer Weise der Sender mehr Rechenkapazität besitzt als der Empfänger. Ähnlich einzuordnen sind dieser Tage aktuelle Kompressionsverfahren „der zweiten Generation“, bei denen der Codierer zunächst versucht, gewisse Strukturen innerhalb des Bildes zu finden und dann diese Information zu übertragen. Auch diese Verfahren zeigen bei hohen Kompressionsraten sehr gute Bildqualitäten.

Wer hierzu vergleichende Bilder studieren will, sei auf [4, 5] verwiesen.

Literatur

- [1] K. Sayood. *Introduction to Data Compression*, Morgan Kaufmann Publishers, Inc., Second Edition, 2000.
- [2] K. Bosch. *Elementare Einführung in die angewandte Statistik*, Vieweg, 1987.
- [3] R. J. Clarke. *Transform Coding of Images*, Academic Press, 1985 u. 1990.
- [4] O. Egger, P. Fleury, T. Ebrahimi, M. Kunt. *High-Performance Compression of Visual Information—A Tutorial Review— Part I: Still Pictures*, Seiten 974–1011 in: Proc. IEEE, Band 87, Nr. 6, Juni 1999. **Hinweis:** Diese Zeitschrift enthält meist empfehlenswerte Übersichtsartikel zu verschiedensten Gebieten der Informatik.
- [5] U. Simon, M. Berndtgen. *Handlich bunt; Kompressionstechniken für Bild- und Videodateien im Vergleich*, Seiten 222 ff. in: c't, Magazin für Computertechnik, Nr. 11, November 1996.

- [6] Al Bovik, Ed., *Handbook of Image and Video Processing*, AP Series in Communication, Networking and Multimedia, AP, 2000.