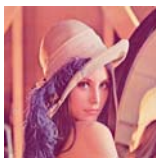
MultiMediaPraktikum

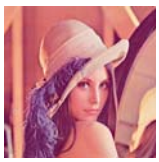
Universität Tübingen Wilhelm-Schickard-Institut für Informatik

Wintersemester 2002/ 2003

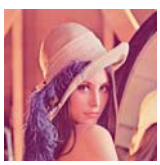
Aufgabe 1 - Bildformate



Marc-Oliver Pahl



David Eißler



Ulrike Schaal

Inhaltsverzeichnis

1. Verschiedene Grafikformate...

bmp	Microsoft Windows Bitmap
gif	Graphics Interchange Format
jpg	joint photographic experts group file interchange format
mac	Macintosh Paint
pct	Macintosh Pict
pcd	Kodak Photo CD
pcx	PC-Paintbrush FileFormat
ras	SUN Raster
tga	Targa Image Format
tiff	Tagged Image File Format
sct	Scitex CT Bitmap
fpx	Kodak Flashpix Bild
wi	Wavelet Compressed Bitmap
dib	Device Independ Bitmap
png	Portable Network Graphic
pf	IBM Pif
fif	Fractal Image Format
...	

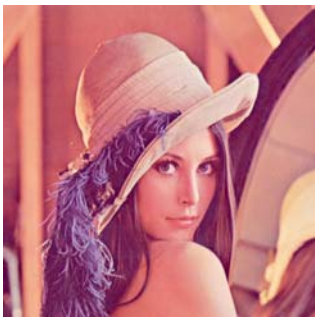
2. Die Bildformate BMP, TIFF, GIF und PNG...

Sowohl Bitmap, als auch Tagged Image File Format, Graphics Interchange Format und Portable Network Graphic sind verlustlose Grafikformate. D.h. anders als z.B. bei JPG kann man aus dem komprimierten Bild das Originalbild zurückgewinnen (bitgleich).

Das TIFF-Format unterstützt die meisten Zusatzinformationen in der Datei.

GIF kann nur Bilder mit 8bit-Farbinformation speichern.

3. Konvertieren Sie mit Hilfe eines Bildverarbeitungsprogramms folgende Beispieldateien in die Formate TIFF, GIF und PNG...



lena

nova

button

	bmp	**tiff**	**gif**	**png**
Farbraum	24bit	32bit	8bit	24bit(-48bit)
Kompressionsverfahren	RLE	RLE, LZW, CCITT Group 3 und 4	LZW	LZW, Huffman
Gut geeignet für	durch RLE für große homogene Flächen	Scanner, Grafikanwendungen, Druckvorstufe	Internet (Alpha-Farbe), LineArt	Internet, große Dateien, Fotos
Ungeeignet für	Internet	Internet	Farbverläufe	LineArt
Vorteile	verlustfrei, voller Farbraum, kostenlos	verlustfrei, CMYK	verlustfrei, Alphakanal, Animationen, Interlacing	verlustfrei, Interlacing
Nachteile	groß	mittelgroß	nur 8bit	bei 24bit groß
Besonderheiten/Sonstiges	„Windows-Standard“	Bildunterschriften und Zusatzinformationen können mitgespeichert werden	interlaced, mehrere Bilder in einer Datei	nicht so verbreitet
Größe der komprimierten Dateien	768kB	638kB	218kB	24bit: 465kB 8bit: 190kB
Sich daraus ergebender Kompressionsfaktor	1:1	1,2:1	3,5:1	1,65:1 4:1

JPG

Erklären Sie den Ablauf einer JPEG-Kodierung...

1. Bildaufbereitung

   Änderung des Farbraums von RGB nach YUV.

   Durch den Übergang von RGB nach YUV werden Farbinformation (UV) und Helligkeitsinformation (Y) getrennt.

   Auf Helligkeitsunterschiede reagiert das Auge wesentlich stärker als auf Farbunterschiede.

2. Bildverarbeitung

   Das macht man sich bei der **Unterabtastung** zu Nutze. Hierbei werden die Farbunterschiede nicht für jedes Pixel sondern nur für mehrere Pixel zusammen gespeichert.

   D.h. bei 4:2:2 (Y:U:V) werden für 4 Pixel 4 Helligkeitsunterschiede (Y) aber nur 2 Farbunterschiede (U,V) gespeichert, bei 4:1:1 entsprechend nur einer.

   X X X X
   Y Y Y Y 4
   U U 2
   V V 2

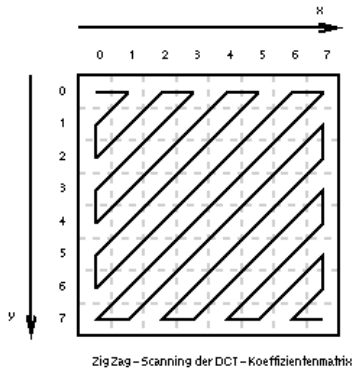
   Als Nächstes wendet man auf **8x8** große Pixel**Blöcke** die **Diskrete Cosinus-Transformation** (DCT) an. Dass die Blöcke gerade 8x8 groß sind hat sich experimentell als beste Größe ergeben, weil dann der Kosten-/ Nutzenfaktor am besten ist (wenig Rechenaufwand).

   Die DCT überführt das Bild vom Ortsraum (Pixel) in den Frequenzraum (Frequenzen von Sinus- und Cosinusfunktionen).

3. Quantisierung

   Durch die DCT wird der Datenanteil aber nicht verringert sondern in der Regel erhöht. Da man höherfrequenten Anteile nicht so stark wahrnimmt, kann man diese im Zuge der **Quantisierung** wegreduzieren (Division). Dadurch stehen jetzt von oben links nach unten rechts vorgehend zunehmend Nullen in der Matrix.

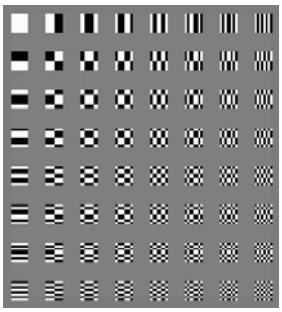
4. Entropiecodierung



Zig Zag – Scanning der DCT – Koeffizientenmatrix

Durch **Zick-Zack-Scanning** stehen diese Nullen dann hintereinander, was ein effektives Packen mithilfe der **Lauflängenkodierung** (Run-Length-Codierung (RLC)) erlaubt. Das anschließende packen mithilfe **variabler Lauflängencodierung** (VLC) **(Huffman)** verringert das Datenvolumen erneut.

Erläutern Sie folgende Abbildung...



Die Abbildung zeigt anschaulich, welche Frequenzen nach der DCT an welcher Stelle in der Matrix stehen.

Also links oben der DC-Anteil (0) und dann die immer höherfrequentigeren AC-Anteile.

Neben dem Baseline-Modus existieren für das JPEG-Verfahren drei weitere Modi. Nennen und erläutern Sie diese.

Progressiv - hier werden nicht pro Einheit (8x8 Block) immer alle Matrixkoeffizienten gespeichert, sondern zuerst die niederfrequenten Anteile (z.B. nur DC) für jeden Block und anschließend immer höherfrequentere. Dadurch ist es möglich, schon ohne das gesamte Bild geladen zu haben, einen Überblick zu bekommen.

Lossless - wie der Name schon sagt komprimiert dieser Mode verlustfrei. Und zwar in dem Sinne, dass dieselben Bits nach der Dekomprimierung herauskommen, die vor der Komprimierung da waren. Deshalb schafft das Verfahren aber auch nur eine Komprimierung im Verhältnis 2:1 und funktioniert auch nur auf echfarbigen Abbildungen sinnvoll. Anstelle der DCT wird ein Prädiktor zum Packen verwendet.

Sequential - der Baseline Modus kodiert so. D.h. von links nach rechts und von oben nach unten werden die Frequenzen immer höher.

Hierarchical - hierbei werden die Bilddimensionen immer halbiert und die Differenzen zur nächsthöheren Auflösungsstufe mit einem der oberen drei Modi kodiert. Dadurch ist es natürlich sehr gut möglich Bilder mit einer der halbierten (bzw. einer geringeren) Auflösung darzustellen.

Im Gegensatz zu GIF oder PNG unterstützt JPEG keine Transparenz. Worin liegt das Problem? Welche Ansätze wären möglich, dennoch eine Transparenz in JPEG zu realisieren?

Beim GIF-Format wird einfach eine Farbe als transparent definiert. Dieser Ansatz kann beim JPG nicht funktionieren, da gerade die genauen Farbinformationen im Zuge der Komprimierung verloren gehen. (Eine Fläche, die im Original einfarbig war, wird im JPG zumeist aus vielen - wenn hoffentlich auch nicht wahrnehmbar - verschiedenen Farben bestehen)

Beim PNG Format gibt es einen Alpha-Kanal (Durchsichtigkeit in % pro Pixel gespeichert). Solch einen Alpha-Kanal könnte man auch im JPG erzeugen und encodieren. Da dieser Kanal aber aus vielen großen gleichfarbigen Flächen (hier durchsichtig) bestehen würde und JPG gerade solche Bilder nur sehr schlecht packen kann ist dies nicht sinnvoll.

Ein sinnvollerer Ansatz, wäre es, einen Alpha-Kanal oder auch nur eine Transparenzmaske seperat im JPG-File zu speichern und zwar mit GIF-Encoding zum Beispiel.

Woran liegt es, dass die Qualität eines JPEG-Bildes schlechter wird, wenn man dieses wiederholt mit dem JPEG-Verfahren komprimiert? Mit welchen Einstellungen kann man diesen zunehmenden Verlust möglichst gering zu halten?

Das JPG-Verfahren ist verlustbehaftet. Und zwar werden gerade die für das Auge möglichst nicht sichtbaren Anteile aus dem Bild gefiltert.

Das bedeutet aber, dass beim dekomprimierten Bild schon diese Informationen rausgefiltert sind und daher beim erneuten Komprimieren nicht wieder wegfallen können. Insbesondere entstehen beim wiederholten Komprimieren Blockartefakte (8x8 Blöcke), weil eben innerhalb immer dergleichen Blöcke immer mehr Information verloren geht.

Man kann versuchen, diesen Verlust möglichst gering zu halten, indem man nur die geänderten Teile des Bildes erneut (mit derselben Einstellung!) komprimiert, oder aber das Bild vor dem Komprimieren unschärft, was die 8x8 Blöcke etwas vermischt. (Das rotieren um 90° und das Abschneiden von 8x8 Blöcken am Rand ist verllustlos möglich)

JpegEncoder

Schauen Sie sich den Java-Quellcode des JPEG-Encoders genauer an. An welcher Stelle geht der Parameter „quality“ ein? Was macht dieser?

Der Parameter wird aus der Eingabe geparst und fließt bei der Erstellung des neuen JPEGEncoders in Zeile 95 von jpeg.java ein:

```
jpg = new JpegEncoder(image, Quality, dataOut);
```

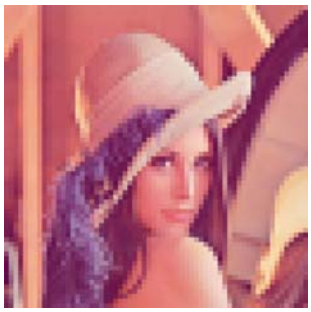
Dort wird in Zeile 75 (jpegencoder.java) mit diesem Parameter eine neue DCT erzeugt:

```
dct = new DCT(Quality); //The quality of the image (0 worst - 100 best)
```

Der Parameter legt fest, durch was die Luminanz- und Chrominanz-Koeffizienten geteilt werden und somit, wie quantisiert wird - also welche bzw. wie viel Information wegfällt.

Geben Sie die vom Encoder verwendeten Quantisierungsmatrizen für Luminanz und Chrominanz bei einer Qualität von 80...

```
Luminanz:
        48      44      42      56      80      101     87      53
        55      77      87      104     111     201     144     67
        63      72      82      123     167     189     158     63
        56      91      111     133     188     259     163     65
        56      100     157     207     216     277     178     68
        63      122     181     192     201     207     153     64
        87      156     175     178     178     163     112     48
        64      113     110     101     99      69      49      24
Chrominanz:
        56      78      105     179     320     251     173     88
        78      123     145     339     444     349     240     122
        105     145     300     492     418     328     226     115
        179     339     492     442     376     296     204     104
        320     444     418     376     320     251     173     88
        251     349     328     296     251     198     136     69
        173     240     226     204     173     136     94      48
        88      122     115     104     88      69      48      24
```



Setzen Sie nun alle Einträge der Quantisierungsmatrizen mit Ausnahme der beiden DC-Werte (1 DC Luminanz, 1 DC Chrominanz) auf unendlich (bzw. 0 als Multiplikator).

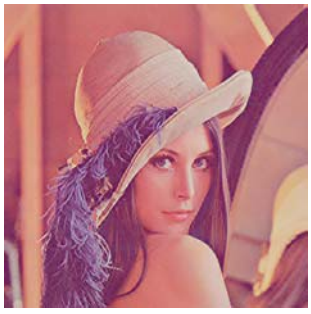
Dadurch werden in jedem 8x8-Block immer nur die Grundschwingungen für Luminanz und Chrominanz im Frequenzraum berücksichtigt, was dazu führt, dass die Blöcke jeweils als einfarbige Fläche dargestellt werden und auch innerhalb des Blockes keinerlei Farb- und Helligkeitsunterschiede sichtbar werden.

Stellen Sie anschließend die AC-Werte wieder her und setzen Sie nun nur die DC-Werte auf unendlich. Kodieren Sie die Bilder mit dieser Einstellung.

Nur Chrominanz DC auf unendlich (0).

Nur Luminanz DC auf unendlich (0).

In diesem Fall werden jeweils Grundfarbe und Grundhelligkeitsverschiebung nicht berücksichtigt. Dadurch erscheint das Bild hauptsächlich in grau (nur noch Farbunterschiede zur verworfenen Grundfarbe im Block) und ohne Helligkeitsunterschiede zwischen den Blökken (nur noch Unterschiede zur nicht berücksichtigten Grundhelligkeit innerhalb der Blöcke).



Setzen Sie nun alle Quantisierungsmatrizen auf einen einheitlichen Wert, zum Beispiel 64. Kodieren Sie nun die Bilder mit dieser Einstellung.

Dadurch werden alle Schwingungen gleich gewichtet. Das führt dazu, dass die hochfrequenten Anteile zu stark und die niederfrequenten Anteile zu schwach berücksichtigt werden.

Helligkeits- und Farbunterschiede innerhalb des Blocks werden daher betont und die Grundhelligkeit und Farbe zu schwach gewichtet. Daher kommt es zu Fehlfarben, was man vor allem bei Werten <65 sieht.

Advanced Imaging - JPEG 2000

Was sind die Vorteile der WT gegenüber der DCT?

Es gibt nicht nur ein paar wenige Grundwellen, wie z.B. Sinus und Cosinus bei der DCT sondern unendlich viele Wavelets.

Sie ist nicht so störanfällig (nur lokale Störungen (Unschärfe), keine Blockbildung).

Der Aufwand wächst nur linear mit der Bildgröße (kleinere Dateien bei gleicher Qualität).

Sowohl bei JPEG als auch bei JPEG2000 wird das Bild in Blöcke zerlegt. Ist dies bei JPEG2000 notwendig? Warum macht man es?

Die Zerlegung in Blöcke ist bei JPEG2000 nicht nötig, man macht sie dennoch, weil so der Zugriff auf Bildteile möglich ist, ohne das gesamte Bild zu bearbeiten (jeder Block ist in sich abgeschlossen kodiert).

Außerdem kann man so Bildteile mit besonders relevanter Information (region of interest (roi)) weniger verlustreich packen als andere.

Informieren Sie sich, in welchen Gebieten außer der Bildverarbeitung Wavelets auch eingesetzt werden. Nennen Sie ein paar Beispiele...

Wavelets haben ein vielfältiges Anwendungsgebiet. Zum Beispiel bei Spracherkennung, in der Mathematik, in der Physik, beim Lösen von Differentialgleichungen, in der Statistik, bei der Erforschung von Fraktalen, bei UMTS, bei mobilem Internet, in der Raumfahrt, in der Medizin, bei der Übertragung in langsamen Kanälen (da progressiv), ...

Erklären Sie die FWT mit Hilfe der Hochpass-, Tiefpassfilter anschaulich!

Vorgehen:

Original:

XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX

calcx:	sortiert:	
SDSDSDSD	SSSSDDDD	
SDSDSDSD	SSSSDDDD	S = Tiefpass (Summe)
SDSDSDSD	SSSSDDDD	D = Hochpass (Differenz)
SDSDSDSD	SSSSDDDD	

calcy:	sortiert:	
ZZZZDDDD	ZZZZDDDD	
OOOODDDD	ZZZZDDDD	Z = Tiefpass (Summe)
ZZZZDDDD	OOOODDDD	O = Hochpass (Differenz)
OOOODDDD	OOOODDDD	

=> Rest zu verarbeiten links oben (Zs)

Zuerst werden die Zeilen des Originalbildes je einmal Tiefpass und Hochpass gefiltert. Aus den beiden neu entstandenen Bildern wird jede zweite Zeile entfernt.

Auf diesen beiden Halbbildern werden nun dieselben Operationen auf die Spalten angewendet.

Das Tiefpass-Zeilen-Tiefpass-Spalten-Bild enthält das Bild in geringerer Auflösung.

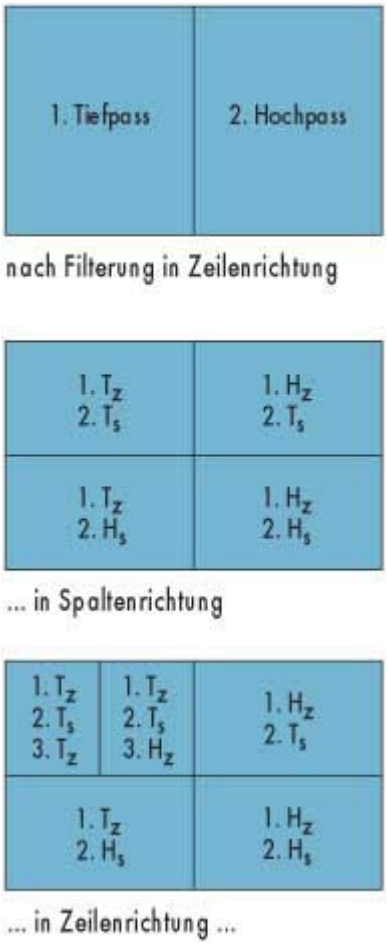
Die Tiefpass-Zeilen-Hochpass-Spalten-Version enthält Differenzinformationen in horizontaler Richtung; die Hochpass-Zeilen-Tiefpass-Spalten-Version in vertikaler Richtung.

Die Hochpass-Zeilen-Hochpass-Spalten-Version enthält Änderungen in diagonaler Richtung.

Die Differenzbilder zeigen gerade die Stellen des Originalbildes an, an denen besonders starke Helligkeits- und Farbunterschiede auftreten.

Das Verfahren wird rekursiv auf die auflösungsreduzierte Version des Bildes angewandt und zwar so lange, bis nur noch ein Pixel übrig bleibt. Da die letzten Rekursionsschritte aber zu keiner relevanten Datenreduktion mehr führen, kann man auch schon früher aufhören, was insbesondere bei nicht viereckigen Bildern und Bildgrößen, die keiner Zweierpotenz entsprechen Anwendung findet.

Das Ergebnis :



Wie funktioniert die Umkehrung des obigen Verfahrens?

Man fügt nun bei der skalierten Version des ursprünglichen Bildes und den dazu gehörigen Differenzbildern jeweils eine Null-Spalte ein. Anschließend wendet man den inversen Hoch- und Tiefpassfilter auf die Spalten an. Die Resultate werden aufsummiert.

Dasselbe wird nun in Zeilenrichtung durchgeführt.

Das Ganze wiederholt man nun so lange, bis man bei der gewünschten Auflösungsstufe bzw. beim Originalbild angelangt ist.

Im Kontext von JPEG2000 fallen oft die Begriffe „multi-resolution“, „random access code stream“ und „region of interest“. Erklären Sie diese...

multi-resolution - da die Bilddimensionen im Zuge der Komprimierung immer halbiert werden, ist es leicht möglich Bilder kleinerer Auflösung zu generieren, indem man einfach nicht bis zum Ende dekomprimiert, sondern bei der gewünschten Auflösung aufhört (falls diese nicht vorhanden ist, kann man z.B. zwischen zwei benachbarten Auflösungsstufen trilinear interpolieren).

random access code stream - durch die Aufteilung in Blöcke ist es möglich auf bestimmte Bildteile relativ schnell zuzugreifen, da immer nur die beteiligten Blöcke dekomprimiert werden müseen und nicht das gesamte Bild.

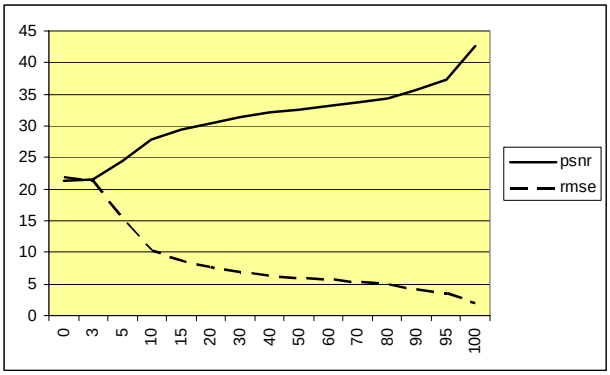
region of interest - durch die Zerlegung in Blöcke ist es möglich jeden Block mit einer anderen Kompressionsstufe zu packen. D.h. interessante Bildteile weniger stark uninteressante stärker.

Vergleich der Bildqualität von JPEG mit JPEG2000

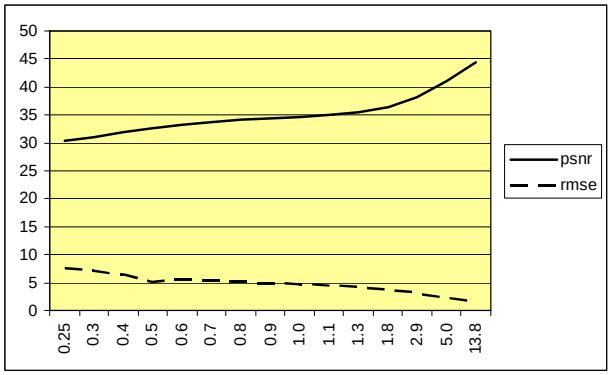
Subjektiv sind vor allem bei niedriger Dateigröße die JPEG2000-Bilder qualitativ deutlich besser als die entsprechenden JPG-Bilder.

Das zeigt sich auch an der Auswertung mit dem ICMP-Programm:

JPEG



JPEG2000



Bei JPEG ist der Signal-Rauschabstand (psnr) bis zur mittleren Qualitätsstufe (die Qualitätsstufen entsprechen sich in der Dateigröße ungefähr) geringer und der gemittelte Bildfehler (rmse) höher als bei JPEG2000. Das heißt bei ungefähr gleicher Dateigröße sind die Ergebnisse deutlich schlechter. Ab etwas über der Mitte trifft dies nicht mehr zu. Das bedeutet, ab da ist es egal, welches Format man verwendet, da JPEG2000 aber auch dort noch leicht besser ist, sollte es sich durchsetzen.

Die Diagramme zeigen nicht ganz, was sie sollten, eigentlich muss man noch folgendes berechnen und auftragen:

$$\mathrm{MSE}(A,B) = \frac{1}{wh} \sum_{0 \le x < w, 0 \le y < h} (A(x,y) - B(x,y))^2$$

$$\mathrm{PSNR}(A,B) = 20 \log_{10} \frac{255}{\sqrt{\mathrm{MSE}(A,B)}}$$