Multimedia

Vorlesung gehalten im WS '97/98

Oliver Vornberger

Fachbereich Mathematik/Informatik
Universität Osnabrück

Inhaltsverzeichnis

1 Einführung **5**

2 Textkomprimierung **13**
2.1 Run length encoding (Lauflängenkomprimierung) 14
2.2 Entropie, Modellierung und Codebäume 15
2.3 Shannon/Fano-Komprimierung 16
2.4 Huffman-Komprimierung (1952) 18
2.5 Adaptive Huffman-Komprimierung 21
2.6 Arithmetische Komprimierung (1982) 25
2.7 LZ-77-Komprimierung (Lempel/Ziv, 1977) 28
2.8 LZ-78-Komprimierung (Lempel/Ziv, 1978) 29
2.9 LZW-Komprimierung (Lempel/Ziv/Welch, 1984) 30

3 Binärbild **33**
3.1 Kompression nach CCITT Gruppe T4 33
3.2 Scanner & OCR 42
3.3 Binärisierung 50

4 Grauwertbilder **53**
4.1 Geometrische Transformationen 53
4.2 Graubildoperationen 54
4.3 Fourier-Transformation 59

5 Farbbilder **65**
5.1 Farbmodelle 66
5.2 Grafikkarte 68
5.3 Farbtabelle 69
5.4 Kompression durch bildbezogene Farbtabelle 71
5.5 Kompression nach JPEG 75
5.6 Fraktale Kompression 84

6 Bilddateiformate **97**
6.1 TIF 97
6.2 Photo-CD 101
6.3 Auflösung 102
6.4 Adobe Photoshop 104
6.5 Apple Quicktime Virtual Realilty 106

6.6 Morphing 108
6.7 Macromedia Flash 109

7 Computergrafik **111**
7.1 Repräsentation 112
7.2 2D-Grafik 114
7.3 3D-Transformation 114
7.4 Projektion 116
7.5 Rendering 118
7.6 Ray Tracing 123
7.7 Caligary trueSpace 124

8 VRML **127**
8.1 Geschichte 127
8.2 Einbettung 128
8.3 Geometrie 129
8.4 Wiederverwendung 132
8.5 Interaktion 134
8.6 Animation 136
8.7 Scripts 138
8.8 Multiuser 140

9 Audio **141**
9.1 Tonhöhe und Lautstärke 141
9.2 Digitalisierung 143
9.3 Creative WaveStudio 146
9.4 Optische Speicher 148
9.5 MPEG-Audio 152
9.6 ISO Aencode/Xaudio 153
9.7 RealAudio 154

10 Sprache **155**
10.1 Akustische Vorverarbeitung 158
10.2 Hidden-Markow-Ketten 159
10.3 Trigramme 161
10.4 IBM ViaVoice 161
10.5 Automatische Sprachübersetzung 161

11 Musik **163**
11.1 Tonsysteme 165
11.2 Tongenerator 169
11.3 MIDI 170
11.4 Roland Masterkeyboard PC-200 173
11.5 Steinberg Cubasis 173
11.6 Microsoft Music Producer 176

12 Video Analog **177**
12.1 Schwarz-Weiß-Fernsehen 177
12.2 Farbfernsehen 178
12.3 Videoaufzeichnung 180
12.4 Time Code 182
12.5 Gold Disk Video Director 183

13 Video Digital **185**
13.1 Videoformate 185
13.2 MPEG 186
13.3 H.261 189
13.4 MPEG-2 190
13.5 Fast FPS 60 190
13.6 Adobe Premiere 191
13.7 Ligos LSX-MPEG-Encoder 192
13.8 Video-on-Demand 193

14 Autorensystem **195**
14.1 Macro Media Director 195

Danksagung

Ich danke ...

Frau Gerda Holmann für sorgfältiges Erfassen des Textes und Erstellen der Grafiken,

Herrn Frank Lohmeyer und Herrn Frank M. Thiesing für ihre engagierte Mitarbeit bei der inhaltlichen und äußerlichen Gestaltung des Textes,

Herrn Axel Hädicke für sorgfältiges Korrekturlesen.

Herrn Viktor Herzog für die Konvertierung des Skripts nach HTML.

HTML-Version
Der Inhalt dieser Vorlesung kann online betrachtet werden unter
`http://www-lehre.informatik.uni-osnabrueck.de/~mm`

Osnabrück, im Mai 1998

(Oliver Vornberger)

Kapitel 1

Einführung

Modewort *Multimedia*

Indizien für Anforderungen des Marktes:

Siemens	sucht	Multimedia Research Scientist
Bertelsmann	sucht	Multimedia-Talente
Activ-Consult	bildet aus zum	Multimedia-Produzenten Multimedia screen designer Multimedia-Programmierer
Spiegelartikel	redet von	Multimedia-Zeitalter Multimedia-Star Multimedia-Unternehmer Multimedia-Experten Multimedia-PC
Neue Osnabrücker Zeitung	bietet an	Multimedia-Einbauküche

Lehmanns führt mehr Bücher mit "Multimedia" im Titel als Bücher mit "objektorientiert" im Titel

DEUTSCHLAND

blikums wurde dagegen gewaltig überschätzt. Überall tun sich die Firmen schwer, genügend Tester zu gewinnen, obwohl die Versuchshaushalte meist das gesamte Equipment und zum Teil sogar die Dienste umsonst bekommen.

Ist die kommende Multimedia-Welt also nur ein Hirngespinst wachstumssüchtiger Manager? Mit Sicherheit nicht: Auch Skeptiker sehen einen gewaltigen Markt – aber weniger bei privaten Konsumenten.

Den großen Durchbruch, glaubt der Technologieexperte Erich Kiefer, erlebt die neue Technik in der Kommunikation der Unternehmen. Kiefer: „Es ist viel billiger, schneller und effektiver, nur noch das zu transportieren, was prinzipiell nicht in Form von Bits bewegt werden kann."

Besonders stark könnten die Infobahnen bei der Telearbeit in das gewohnte Leben vieler Menschen eingreifen. Rund 70 Prozent aller Arbeitsplätze in Deutschland, schätzt der Bonner Multimedia-Unternehmer Thomas Garmhausen, sind nicht an einen bestimmten Standort gebunden. Dank Computer- und Telekommunikationstechnik könnten deshalb viele Menschen genausogut zu Hause oder an einem beliebigen anderen Standort arbeiten.

Bei IBM arbeiten schon heute mehr als 5000 Angestellte, darunter knapp 400 in Deutschland, ganz oder teilweise von zu Hause aus. Die Auslagerung rechnet sich: Da sich 6 bis 8 Angestellte einen Platz im Büro teilen, will die Firma in den nächsten Jahren 20 Prozent ihrer Büroräume streichen und dadurch Kosten in zweistelliger Millionenhöhe sparen. Langfristig, glauben Experten, lassen sich mit jeder in neue Technik investierten Mark zwei Mark an Büromieten sparen.

Wenn 1998 die Monopole der Telekom fallen, werden Dutzende von privaten Anbietern auf den Markt drängen und die Preise drücken. Dann wird sich, davon ist Multimedia-Experte Garmhausen fest überzeugt, „die Virtualisierung von Arbeitsplätzen auch in Deutschland zügig durchsetzen".

Seit Monaten schon läßt der gewiefte Strategie Kirch Mitarbeiter über speziel-

Telearbeit wird das Leben vieler Menschen verändern

le Nachrichtenkanäle für geschlossene Zirkel nachdenken. Ein Ergebnis dieser internen Planungen: Kirchs „dBox" könnte beispielsweise bei der Personalschulung des Autokonzerns BMW zum Einsatz kommen. Von der Zentrale in München aus, so Kirchs Experten, ließen sich über Satellit alle Werkmeister in den deutschen Niederlassungen sowie benötigte Mitarbeiter in ihren Wohnungen verbinden. Dann könnten neue Maschinen oder Montageanleitungen live erklärt werden. Die Kostenvorteile liegen auf der Hand.

Der Streit unter Experten geht deshalb längst nicht mehr um die Frage, ob sich Multimedia durchsetzen wird. Das Problem für die Elektronikindustrie ist, wie schnell sich die neuen Techniken etablieren.

Viel hängt davon ab, wie gut es der Industrie gelingt, die Geräte narrensicher und benutzerfreundlich zu machen. Denn Multimedia, sagt der Münchner Medienexperte Rüdiger Funiok, „muß auch von technikungewohnten Menschen leicht beherrschbar sein".

Noch sind die Computerfirmen, aber auch die Elektronikbranche, davon weit entfernt. Nicht einmal 40 Prozent aller Besitzer von Videorecordern, glauben Kenner der Szene zu wissen, können ihr Gerät selber programmieren. Gerade einmal 10 Prozent aller Funktionen eines Komforttelefons, so lautet ein anderer Erfahrungswert der Branche, werden wirklich genutzt.

„Euphorie ist fehl am Platz", meint deshalb der Saarbrücker Marketingexperte Joachim Zentes. Multimedia werde sich „eher langsam entwickeln".

Doch Medienunternehmer wie Leo Kirch oder der Time-Warner-Chef Gerald Levin wollen sich von den Skeptikern nicht das Geschäft ausreden lassen. Sie setzen darauf, daß auch bei anderen Erfolgsprodukten, wie Telefon, Homecomputer oder Handy, zunächst kein Bedarf gesehen wurde.

Levin bemüht deshalb gern die Geschichte. Und die, behauptet der Time-Warner-Chef, „zeigt, daß die Konsumenten noch nie wußten, was sie eigentlich wollten – bis sie die neuen Angebote selbst kennenlernten".

Spiegelartikel *Das Ding der Zukunft* vom 21.8.95

J.: 7 St.
t. 1.07x1.55,
<1.55, 0.99x
Waschbecken
n; Küppers-
le, 3 J.; Russ-
l; 15 Teile
ıg zus. 120
belsäulenkis-
ihle „Happy"
utositz, 0-9
neu; AEG-Ge-
l, voll funkti-
d. Babyklei-
05742/2603
, rustikal, mit
und 2 weitere
ısammen 100
619
ün, m. Kühl-
schmaschine,
ch einzeln,

echts, mit
tabzughaube,
./9276
n, sehr gut er-
700 DM), für
enschoner +
M. Flurgarde-
M. ☎ 05491/

ßsack (grün),
n-Markenklei-
Umstandsmo-
mmelsheim).

stikal, 3,16 m, günstig zu verkaufen. ☎ 05422/49022

Mountainbike, 18-Gang, nur für Berge, 250 DM, zu verkaufen. ☎ 05426/2393 ab 16.30 Uhr

Mountainbike, 21-Gang, Deore LX, Kettenschaltg., Topzust., viele Extr., Typ Univega, NP 1999 DM, Vhb. 1100 DM ☎ 05409/4414.

Multimedia-Einbauküche, mit Double-Speed Microwelle u. 1.2 GBYTE Kühlschrank, Preis Vhs. ☎ 0541/598913

Museumsstück: Mooreichentruhe (1680/sehr präzise Schätzung) an Kenner und Sammler ausgewählter Raritäten zu verkaufen. ☎ 0541/29841

NAD Verstärker, 150 Watt, Amplifier 3100, n.w. 700 DM, NAD Tuner 4100, n.w. 450 DM, CD-Player AD-9020 Fisher, 1 J. alt 200 DM, Boxen B+W DM 630, n.w. 850 DM, zusammen 2100 DM. ☎ 05407/39309

Nerzmantel m. Kappe, Gr. 40/42, neuw., wegen Todesfall günstig abzugeben. ☎ 05409/1213

Notebook Intel 386 mit DOS 6.2 + Windows 3.1. Computer Drucker, Tintenstrahl, Star-jet-SJ 48. Panasonic elektr. Thermo Schreibmaschine ☎ 0541/49151

Orgel, elektr., 2manualig, „Farfisa 255 R Partner", Akkordeon,

ter Zustand
☎ 05409/1
Rhododendre
50-60 cm
☎ 0541/78
Rundecke mit
50 DM. ☎ 0
Räucherschra
☎ 05435/6
Rückentrageg
Monate,
gut erhalten
Sandsteinkam
Brother-Stri
DM Vhb. ☎
Satellitenanla
zum Drehe
mit Receive
999 DM, ☎
Scall-Empfänc
blau, noch
7/96, Vhb.
201575
Schreibtisch,
0,82 m, 7
DM. ☎ 054
Schlafzimmer
ben. ☎ 054
Schlafzimmer
bett, 6türig
Nachtschrä
☎ 0541/57
Schlafzimmer
schwarz,
franz. Bett
schwerer, n
mertisch,
Eiche NP 4

Schnäppchenmarkt der Neuen Osnabrücker Zeitung vom 20.10.95

Bücher mit *Multimedia* im Titel:

Andleigh & Thakrar	Designing Multimedia Systems	[PreHal]
Aston & Schwarz	Multimedia: Gateway to the Next Millennium	[APP]
Barrett & Redmond	Contextual Media. Multimedia and Interpretation	[MIT]
Blattner & Dannenberg	Interactive Multimedia Computing	[AddWes]
Buford	Multimedia Systems	[AddWes]
Chorafas	Intelligent Multimedia Databases	[PreHal]
Dodds	Digital Multimedia Cross-Industry Guide	[Butwor]
Earnshaw & Vince	Multimedia Systems and Applications	[AcaPre]
Eissa	Multimedia Production Guide	[Hayden]
Encarnacao & Foley	Multimedia. System Architectures and Applicat.	[Spring]
Fickert	Multimedia interaktiv	[Vieweg]
Fisher	Multimedia Authoring	[APP]
Floy	IBM Multimedia Handbook	[Brady]
Fluckiger	Understanding Networked Multimedia	[PreHal]
Gibbs & Tsichritzis	Multimedia Programming	[AddWes]
Gillmaier	Mac Sampler IV. Multimedia für den Mac	
Hasebrock	Multimedia Psychologie	[Spekru]
Heller	Putting Multimedia	[McGraw]
Jeffcoate	Multimedia in Practice	[PreHal]
Jennings	Discover Windows 3.1 Multimedia	[Que]
Kappe	Vernetzte Multimediasysteme	[BI]
Klute	Multimedialer Hypertext im Internet	
Lawton & al	Developing Multimedia Applications Under OS/2	[Wiley]
Magnenat-Thalmann	Virtual Worlds and Multimedia	[Wiley]
Meister	Multimedia-Anwendungen auf PC & Apple Macintosh	[Franz]
Meissner	Digitale Multimediasysteme	[Tech]
Messina	Was ist Multimedia?	[Hanser]
Meyer-Wegener	Multimedia-Datenbanken	[Teub]
Müller	Der Multimedia PC	[Vieweg]
Niegemann & Freibichler	Hypertext und Multimedia	
Nielsen	Multimedia and Hypertext	[AcaPre]
Niggl	Multimedia für den Mac	[ITP]
Noll	Musik-Programmierung. MIDI, C und Multimedia	[AddWes]
Ozer	Video Compression for Multimedia	[APP]
Plass	Multimedia	[VDE]
Rathbone	Multimedia für (Dumme) Anfänger	
Reinhard	WHO is WHO in Multimedia [Spring]	
Rimmer	Multimedia Programming for Windows	[Wind]
Rosch	Multimedia Bible	[Sams]
Rosenberg	A Guide to Multimedia	[NRP]
Shaddock	Multimedia Workshop	[Tewi]
Siemoneit	Multimedia. Präsentationen planen	
Steinmetz	Multimedia-Technologie	[Spring]
Turlington	Walking the WWW:A Listings Guide for Multimedia	
Tway	Multimedia In Action	[APP]
Wodaski	Multimedia Madness	[Sams]
Wolfgram	Creating Multimedia Presentations	[Que]
Wratil & Schwampe	Multimedia für Video und PC	[M & T]
Yager	Multimedia Production Handbook	[APP]
Zeller	Multimedia. Technik und Anwendung	

Ursprung des Begriffs *Multimedia*:

Multi- [lat.: viel], als Präfix

Medium [lat.: das in der Mitte Befindliche], allgemein Mittel, vermittelndes Element, insbesondere (in der Mehrzahl) Mittel zur Weitergabe und Verbreitung von Information durch Sprache, Gestik, Schrift und Bild
`Meyer's Enzyklopädisches Lexikon, Band 15, Mannheim 1975`

Data Becker:	Alles über Bild und Ton
Volkshochschule:	Bild und Ton — was ist das? Mo, Mi, Fr, 20:15 bitte Wolldecke mitbringen
Vorlesung:	Digitalisierung und Komprimierung von audiovisuellen Daten Manipulation mit Windows-basierten Werkzeugen

Kreide & Folie
Theorie & Praxis
DCT & Keyboard

Was soll diese Vorlesung/dieses Skript leisten?

nicht:	Wie baue ich ein CD-Rom-Laufwerk ein?
sondern:	Welcher Algorithmus verbirgt sich hinter der Bildverarbeitung "Kanten schärfen"?
nicht:	Wie programmiert man eine gute Multimedia-applikation? (Style Guide)
sondern:	Wie formuliert man mit einem Autorensystem ein Drehbuch?

Medium	**Datenmenge**	**Komprimierung**	**Software**
Text (*)	Fontgröße 8×8 Pixel pro Zeile $640/8 = 80$ Zeichen pro Spalte $480/8 = 60$ Zeichen pro Zeichen 1 Byte $\Rightarrow$ 4800 Bytes	run length encoding Huffman (1:2) PKZIP (1:3)	
Bild (*)	$640 \times 480 \times 3 \Rightarrow$ 900 KByte	Fax-Komprimierung (1:10) Colortable JPEG (1:15) Fraktale Kompression	OCR-Software Photoshop 3.0
Grafik	500 Geraden mit Anfangspunkt 20 Bit Endpunkt 20 Bit Attribut 8 Bit 500×48 Bits $\Rightarrow$ 3 KByte		Caligari TrueSpace: 3 D-Editor Rendering Raytracing VRML Animation
Musik	2 Kanäle mit 44.1 KHz abgetastet mit 16 Bit quantisiert $2 \times 44.100 \times 16 = 1.4$ MBit/sec = 10 MB/min (CD-Player schafft 1.5 MBit/sec)	audio mpeg liefert 384 KBit/sec $\Rightarrow$ 384 KBit : 1.4 MBit = 1:3.5	SB-16-Studio Wavetable-Tool
Sprache	1 Kanal mit 8 KHz abgetastet mit 8 Bit quantisiert $8000 \times 8 = 64$ KBits/sec = 480 KByte/min		IBM Voice Type
Midi	bei Schlagzahl 120 $\Rightarrow$ 30 Takte pro Min pro Takt etwa 10 Ereignisse ein Ereignis (Ton, Aktion) = 3 Bytes 16 Spuren $\Rightarrow 16 \times 30 \times 10 \times 3 = 14.4$ KByte/min	bereits komprimiert bzgl. Musik gilt $\Rightarrow$ 14.4 KB:10 MB = 1:700	Roland Midi Tastatur Steinberg Cubasis Sequenzer
Video	PAL verlangt 625 Zeilen mit 833 Punkten Helligkeit wird mit 13.5 MHz abgetastet 2 Farbdifferenzen werden mit je 6.75 MHz abgetastet, mit 8 Bit quantisiert $(13.5 + 6.75 + 6.75) \times 8 = 216$ MBit/sec = 1.54 GByte/min $640 \times 480 \times 3 \times 25 \times 60 = 1.31$ GByte/min	mpeg liefert 25 MB/min $\Rightarrow$ 25 MB:1.54GByte = 1:60	linearer Schnitt mit Gold Disk Video Director 2.0 Digitalisierung mit FAST FPS/60 nach MJPEG nichtlinearer Schnitt mit Adobe Premiere 4.0 Komprimierung mit LSX-MPEG-Encoder
Multimedia			Macro Media Director

(*) basierend auf VGA-Auflösung 640×480 Pixel

Literatur:

Wolfgang Abmayr:
Einführung in die digitale Bildverarbeitung
Teubner Verlag, 1994. 301 Seiten, DM 68,-

Michael Barnsley, Lyman Hurd:
Bildkompression mit Fraktalen
Vieweg Verlag, 1996, 232 Seiten, DM 98,-

Wolf Dietrich Fellner:
Computergraphik
BI, 1992, 418 Seiten, DM 68,-

Klaus Holtorf:
Das Handbuch der Grafikformate
Franzis Verlag, 1996, 78,- DM, incl. CD-ROM

Heiner Küsters:
Bilddatenkomprimierung mit JPEG und MPEG
Franzis-Verlag, 1995, 264 Seiten, 78,- DM, incl. CD

Hansgeorg Meissner:
Digitale Multimediasysteme.
Verlag Technik Berlin, 1994, 187 Seiten, DM 49,90

Robert Rockwell, Jörg Kloss, Kornel Szabo:
VRML 97
Addison-Wesley, 1997, 350 Seiten, DM 89.90

Hans-Jürgen Schlicht:
Bildverarbeitung digital
Addison-Wesley Verlag, 1995, 358 Seiten, 79,90 DM, incl. CD

Wolfgang Schlichter, Oliver Meissel:
Soundverarbeitung mit MIDI
Franzis Verlag, 1994, 325 Seiten, 78,- DM, incl. CD

Rainer Steinbrecher:
Bildverarbeitung in der Praxis.
Oldenbourg Verlag, 1993. 321 Seiten, 68,- DM

Ralf Steinmetz:
Multimedia-Technologie, Einführung und Grundlagen
Springer Verlag, 2. Auflage 1995, 591 Seiten, 89,- DM

Kapitel 2

Textkomprimierung

Komprimierung von Bild und Ton darf verlustbehaftet sein, da bereits bei der Digitalisierung ein Teil der ursprünglichen Daten verlorengeht.

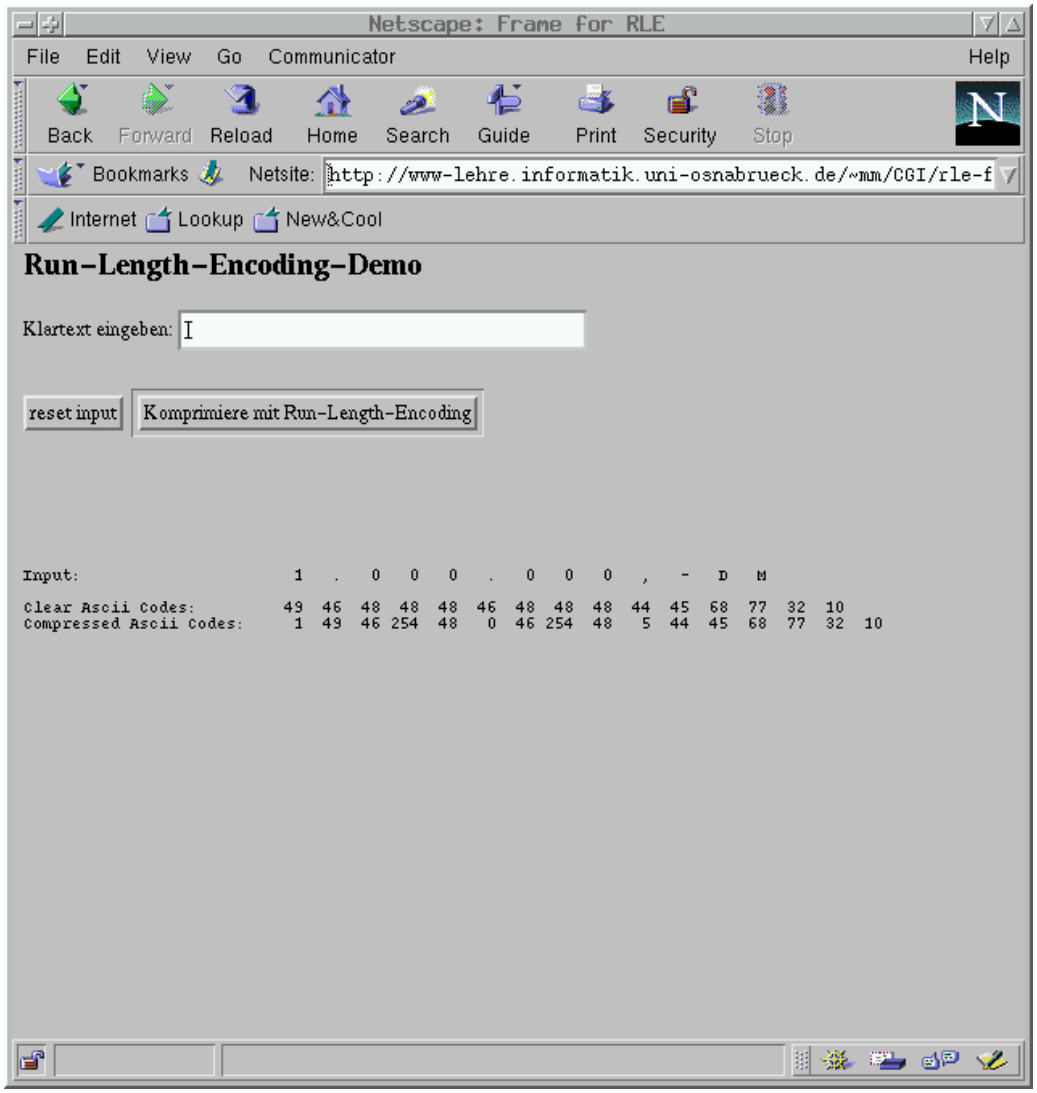
Komprimierung von digital abgelegtem Text soll verlustfrei sein. Es gibt

- Run length encoding
  überträgt eine Folge von identischen Zeichen als Paar <Anzahl, Zeichen>.
- statistische Verfahren
  gehen von einer Wahrscheinlichkeitsverteilung auf dem Eingabealphabet aus,
  verwenden variable length encoding, d.h. häufige Zeichen werden mit wenig Bits, seltene Zeichen mit viel Bits codiert.
  Beispiele: Shannon/Fano, Huffman, arithmetische Codierung.
- tabellengesteuerte Verfahren
  bauen Tabelle mit gelesenen Strings auf und übertragen Verweise in diese Tabelle.
  Beispiele: LZ77, LZ78, LZW.

2.1 Run length encoding (Lauflängenkomprimierung)

$x < 128$: übernimm die folgenden $x + 1$ Zeichen
$x \geq 128$: das nächste Zeichen kommt $257 - x$ mal
$\Rightarrow$ maximal 128 Zeichen übernehmen
maximal 128 Kopien eines Zeichens

```
Zeichenkette  1.000.000,-DM
ASCII-Codes   49  46  48  48  48  46  48  48  48  44  45  68  77
Komprimiert    1  49  46 254  48   0  46 254  48   3  44  45  68  77
```



Beispiel in Ausführung

2.2 Entropie, Modellierung und Codebäume

Entropie (Thermodynamik) =
Maß für Informationsgehalt einer Meldung, bezogen auf eine Wahrscheinlichkeitsverteilung =
$-\log_2 \underbrace{(P(x))}_{\text{Wahrscheinlichkeit für Zeichen } x} = \log_2(1/P(x))$

z.B. 8 Zeichen gleichverteilt
$P(x) = \frac{1}{8} \Rightarrow$ Entropie = 3 Bit pro Zeichen
z.B. 1 Zeichen mit $P(x) = \frac{7}{8} \Rightarrow Entropie = 3 - \log_2\ 7 = 0.2$

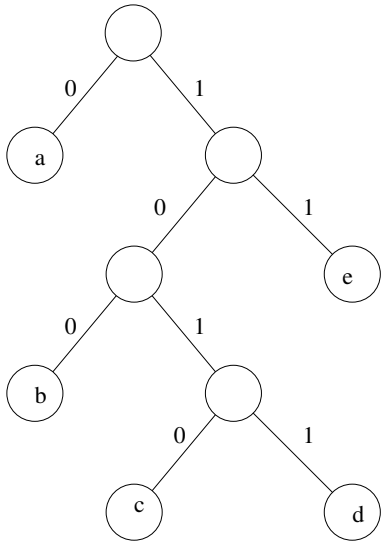
7 Zeichen mit je $P(x) = \frac{1}{56} \Rightarrow$ Entropie $= \log_2\ 56 = 5.8$

Wahrscheinlichkeiten werden bei Modellen höherer Ordnung kontextbezogen definiert, z.B. in Abhängigkeit vom Vorgängersymbol.

Im C-Source gilt $P(newline) = \frac{1}{40}$
$P(newline \text{ hinter } '\}') = \frac{1}{2}$

Bei *variable length encoding* darf kein Code Anfangsstück eines anderen Codes sein (präfixfrei).

Gut durch Baum zu repräsentieren:
Symbol $\cong$ Blatt
Code $\cong$ Kantenbeschriftung auf dem Weg zum Symbol



Symbol	Code
a	0
b	100
c	1010
d	1011
e	11

2.3 Shannon/Fano-Komprimierung

Entwickelt 1960 bei Bell Labs/MIT.

Konstruiere den Code-Baum *top-down*:

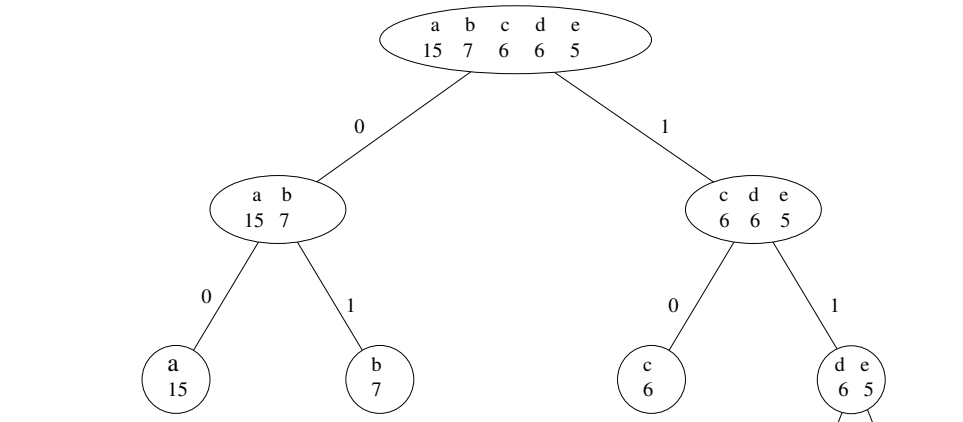
```
Bilde Wurzel mit Liste aller Symbole,
            gewichtet mit Haeufigkeiten

While Blaetter-nicht-einelementig do
   Sei L die Liste eines mehrelementigen Blattes
   Bilde zwei etwa gleichgewichtige Listen L0, L1
   und haenge sie als Soehne an Liste L
      (Kanten erhalten 0/1)
end;
```

Zusätzlich zur codierten Nachricht muß der Baum übertragen werden.

Beispiel: Die Zeichen `a, b, c, d, e` sind in einem Text wie folgt verteilt:

Symbol	Häufigkeit	$P(x)$	$-\log_2(P(x))$	Informationsgehalt
a	15	0.38	1.38	20.67
b	7	0.18	2.48	17.35
c	6	0.15	2.70	16.20
d	6	0.15	2.70	16.20
e	5	0.13	2.96	14.82
	39	1.00		85.25

Nach Anwendung des Algorithmus entsteht folgender Codebaum:



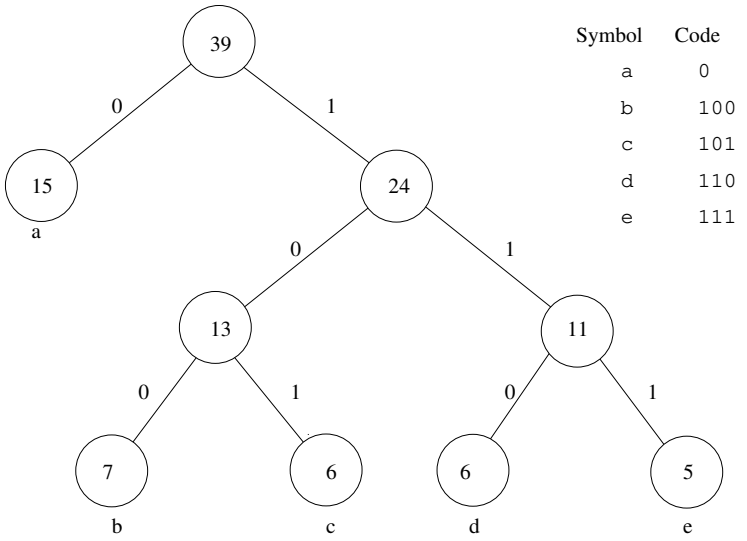
Symbol	Code
a	00
b	01
c	10
d	110
e	111

Die Nachrichtenlänge beträgt 89 Bit, der Informationsgehalt beträgt 85.25 Bit.

2.4 Huffman-Komprimierung (1952)

Konstruiere den Code-Baum *bottom-up*.

```
Bilde Wald mit Blaettern mit Symbolen, gewichtet mit Haeufigkeiten
While noch_kein_Baum
   Seien L0, L1 die beiden Wurzeln mit kleinstem Gewicht
   Bilde Vater L mit Soehnen L0, L1
      (Kanten erhalten 0/1, Vater erhaelt Summe der Sohngewichte)
end;
```



Symbol	Code
a	0
b	100
c	101
d	110
e	111

Nachrichtenlänge beträgt 87 Bit.

Satz: Huffman nie länger als *Shannon-Fano* !
$\cong$ *Unix-utility pack*, komprimiert auf 60 % bei Textfiles.

Verteilung der ASCII-Zeichen im Spiegelartikel *Multimedia*

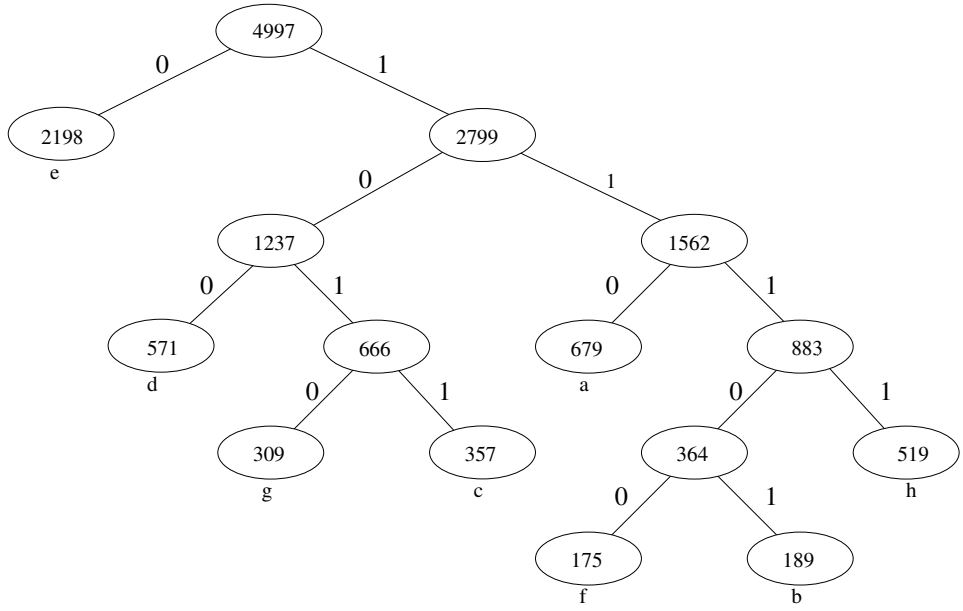
302	↩	2067	␣	1	!	56	\	1	’
9	(| 9 |)	163	,	71	-	135	.
4	/	42	0	16	1	13	2	11	3
11	4	13	5	6	6	5	7	6	8
11	9	20	:	1	;	2	?	60	A
57	B	24	C	87	D	40	E	52	F
40	G	31	H	23	I	16	J	58	K
20	L	86	M	31	N	10	O	49	P
1	Q	21	R	80	S	51	T	19	U
31	V	35	W	1	Y	27	Z	2	[
2	]	679	a	189	b	357	c	571	d
2198	e	175	f	309	g	519	h	1099	i
14	j	209	k	557	l	346	m	1392	n
407	o	127	p	9	q	969	r	636	s
819	t	482	u	94	v	154	w	18	x
17	y	139	z	1	Ä	1	Ü	31	ß
68	ä	35	ö	67	ü				

Häufigkeit einiger Buchstaben im Spiegelartikel *Multimedia*

Symbol	Anzahl	$P(x)$	$-\log_2(P(x))$	Informationsgehalt
a	679	0.14	2.88	1955.23
b	189	0.04	4.72	892.95
c	357	0.07	3.80	1359.12
d	571	0.11	3.13	1786.94
e	2198	0.44	1.18	2604.35
f	175	0.04	4.83	846.24
g	309	0.06	4.02	1240.75
h	519	0.10	3.27	1695.70
	4997			12381.28

= 82 % von 14991 (bei 3 Bit pro Zeichen)

Huffman-Code-Baum für einige Buchstaben im Spiegelartikel *Multimedia*

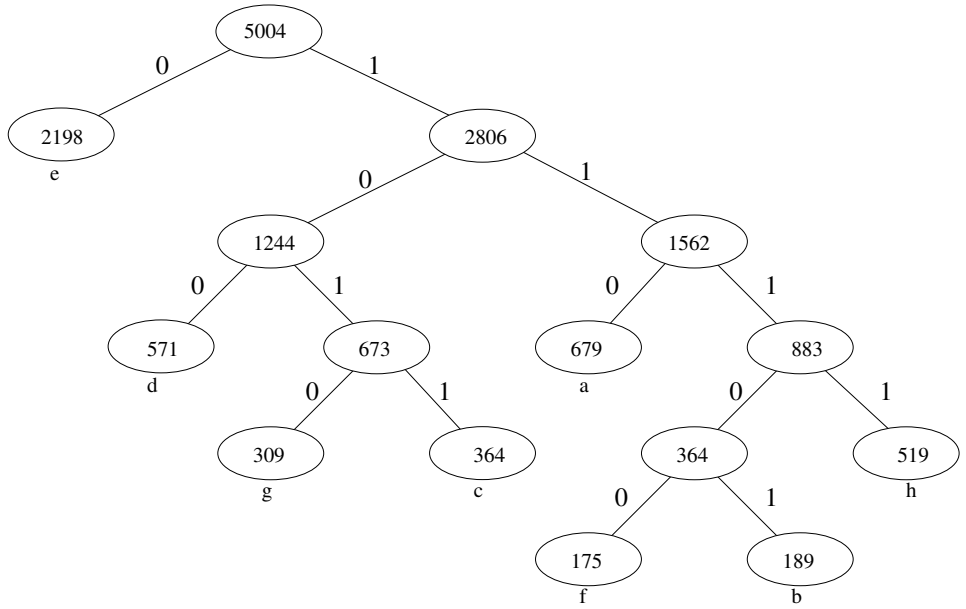


Symbol	Anzahl	Code	Codelänge
a	679	110	2037
b	189	11101	945
c	357	1011	1428
d	571	100	1713
e	2198	0	2198
f	175	11100	875
g	309	1010	1236
h	519	1111	2076
		12508	= 83 % von 14991

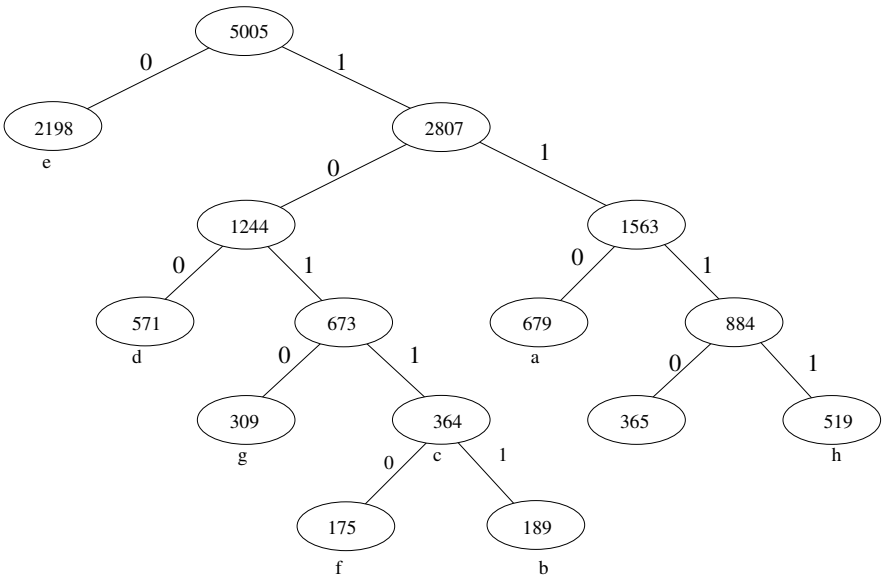
2.5 Adaptive Huffman-Komprimierung

Die relative Ordnung der Symbolgewichte bestimmt die Baumstruktur. Beim Lesen eines Zeichens wird der Zähler im Blatt erhöht, und die Gewichte der Vorfahren werden modifiziert. Ändert sich die relative Ordnung, so wird der Verursacher (mit Gewicht $w+1$) mit dem "letzten" Knoten mit Gewicht w getauscht (incl. Teilbäume).

Z.B. 7 weitere c würden ergeben:



Ein weiteres c würde zu einem Tausch mit dem Vater von f und b führen:



Die neue Code-Tabelle

Symbol	Code
a	110
b	10111
c	1110
d	100
e	0
f	10110
g	1010
h	1111

Der Baum wird nicht übertragen, sondern von Sender und Empfänger initialisiert und dynamisch verändert.

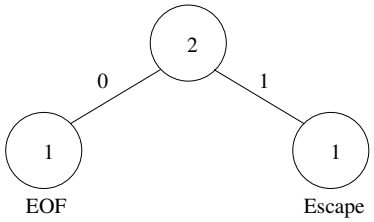
Komprimieren:

```
initialisiere_baum;
repeat
   x := get_char();
   y = codiere(x);
   aktualisiere_baum(x)
   put_bits(y)
until x = EOF
```

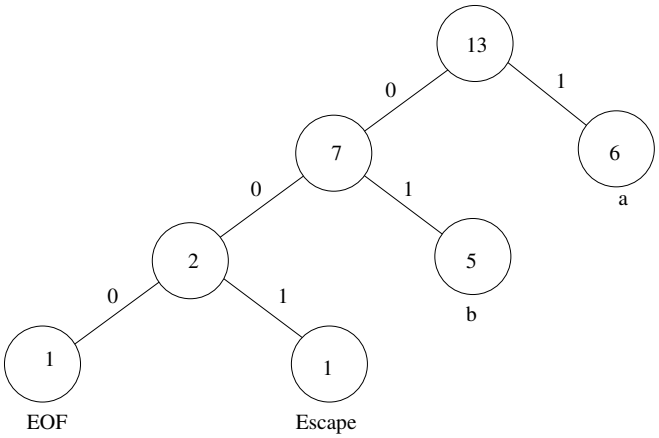
Dekomprimieren:

```
initialisiere_baum;
repeat
   y := get_bits();
   x = decodiere(y);
   aktualisiere_baum(x);
   put_char(x)
until x = EOF
```

1. Möglichkeit zur Initialisierung:
Initialisiere Baum mit Zähler = 0 für jedes mögliche Zeichen.
Nachteil: alle Codes haben zunächst Länge 8, schrumpfen nur langsam.
2. Möglichkeit zur Initialisierung:
Beginne mit leerem Baum, trage nur Symbole ein, die vorkommen. Problem: beim ersten Auftreten kann Symbol nicht codiert werden, da es noch nicht im Baum ist.
Lösung: Bei neuem Zeichen sende Escape-Code + ASCII-Darstellung des Zeichens. Zur Initialisierung des Baumes tragen Sender und Empfänger zwei Zeichen ein:



Nach Lesen des Strings `aabaababababb`:



Nach Einlesen von `abcdefghijklmno`:

a	1	0001
b	1	0010
c	1	0011
d	1	0100
e	1	0101
f	1	0110
g	1	0111
h	1	1000
i	1	1001
j	1	1010
k	1	1011
l	1	1100
m	1	1101
n	1	1110
o	1	1111

Nach Einlesen von `eeeeeeee`:

e	9	0
o	1	1000
a	1	10011
b	1	10100
c	1	10101
d	1	10110
f	1	10111
g	1	11000
h	1	11001
i	1	11010
j	1	11011
k	1	11100
l	1	11101
m	1	11110
n	1	11111

Nach Einlesen von 12 mal `abcdfghijklmno`:

o	13	000
e	9	00101
a	13	0011
b	13	0100
c	13	0101
d	13	0110
f	13	0111
g	13	1000
h	13	1001
i	13	1010
j	13	1011
k	13	1100
l	13	1101
m	13	1110
n	13	1111

2.6 Arithmetische Komprimierung (1982)

Nachteil von Huffman: nur ganzzahlige Codelängen.
Jetzt: ordne den Symbolen disjunkte Teile des Intervalls $[0..1[$ zu:

	$P(x)$	unten	oben
a	0.14	0.00	0.14
b	0.04	0.14	0.18
c	0.07	0.18	0.25
d	0.11	0.25	0.36
e	0.44	0.36	0.80
f	0.04	0.80	0.84
g	0.06	0.84	0.90
h	0.10	0.90	1.00

```
unten   := 0.0;
oben    := 1.0;
bereich := 1.0;

repeat

   x := get_char();

   unten := unten + bereich * untere_grenze(x);
   oben  := unten + bereich * obere_grenze(x);
   bereich := oben - unten

until x = EOF;

putstring(unten); /* Nachricht */
```

Beispiel für "Chef"

	unten	oben	bereich
c	0.18	0.25	0.07
h	0.243	0.25	0.007
e	0.24552	0.2486	0.00308
f	0.247984	0.2481072	0.0001232

Nachricht: 0.247984 (18-Bit Unsigned Integer)
(Huffman: 1011 1111 0 11100 = 14 Bit)

arithmetische Dekomprimierung

```
getstring(y);
repeat
   x := suche_passendes_intervall(y);
   put_char(x);
   bereich := obere_grenze(x) - untere_grenze(x);
   y := (y - untere_grenze(x))/bereich;
until x = EOF
```

Beispiel für Nachricht `y = 0.247984`

Intervall	Symbol	bereich	(y - untere Grenze)/bereich
0.18-0.25	c	0.07	0.9712
0.90-1.00	h	0.10	0.7120
0.36-0.80	e	0.44	0.8000
0.80-0.84	f	0.04	0.0000

Rechenintensiver als Huffman.
Zur Implementation verwendet man Integer-Arithmetik.
Identische, führende Ziffern bei unterer und oberer Schranke können bereits ausgegeben und dann entfernt werden.

`'hhhhhh'` arithmetisch komprimiert ergibt
`0.9 + 0.1 * 0.9 + 0.9 * 0.01 + 0.9 * 0.001 + ... =`
`0.999999` = 20 Bit Integer

`'hhhhhh'` Huffman komprimiert
ergibt `1111 1111 1111 1111 1111 1111` = 24 Bit

Beispiel: Zwei Zeichen haben die Wahrscheinlichkeiten `0.9` bzw. `0.1`

	unten	oben
a	0.0	0.9
b	0.9	1.0

Codierung von `aaaaaaab` ergibt:

	unten	oben	bereich
a	0.0	0.9	0.9
a	0.0	0.81	0.81
a	0.0	0.729	0.729
a	0.0	0.6561	0.6561
a	0.0	0.59049	0.59040
a	0.0	0.531441	0.531441
a	0.0	0.4782969	0.4782969
b	0.4304672	0.4782969	0.0478297

Wähle als Nachricht eine Zahl aus dem letzten Intervall, z.B. `0.45`. Also werden 8 Zeichen mit 6 Bit übertragen.

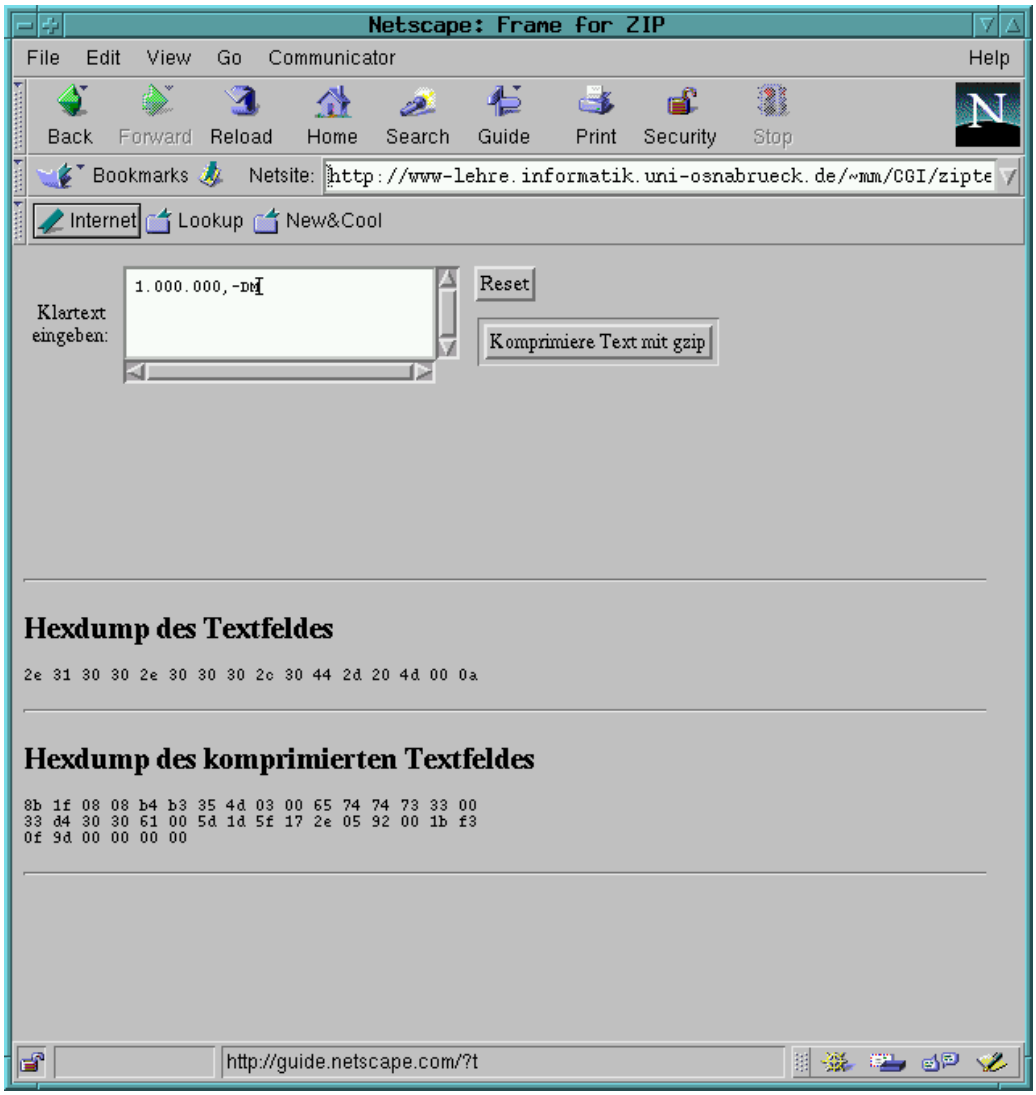
Mark Nelson:
$P('0') = \frac{16382}{16383}$ $P('1') = \frac{1}{16383}$

`100.000` mal `'0'` ergibt arithmetisch komprimiert 3 Bytes!
nach Huffman `100.000 Bits` = 12500 Bytes)

2.7 LZ-77-Komprimierung (Lempel/Ziv, 1977)

Der Kompressionsalgorithmus baut eine Tabelle auf mit Zeichenketten, die sich in einem Ausschnitt aus dem zuvor eingelesenen Eingabestrom befanden.

`gzip` und `pkzip` sind moderne Varianten.

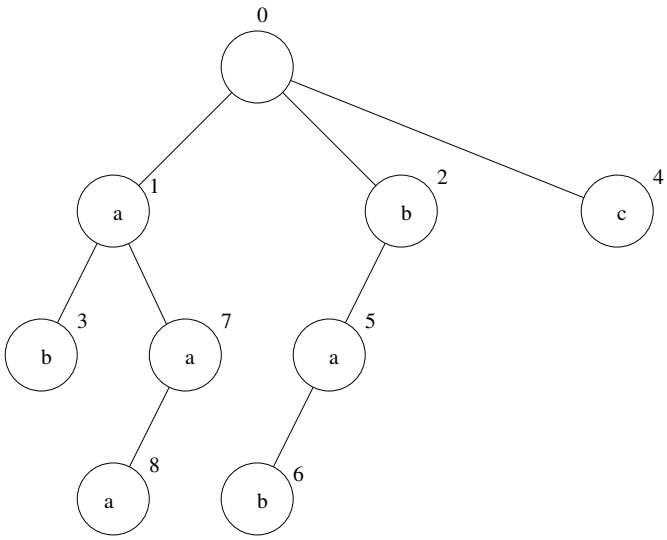


Beispiel der gzip-Textkomprimierung

2.8 LZ-78-Komprimierung (Lempel/Ziv, 1978)

Der Kompressionsalgorithmus baut einen Baum auf und erzeugt als Ausgabe eine Folge von Token. Jedes Token besteht aus einer Adresse im Baum (d.h. Verweis auf einen Knoten) und einem Zeichen. Das Token repräsentiert den String, der dem Weg von der Wurzel bis zum referierten Knoten entspricht, verlängert um das Zeichen.
Sender und Empfänger bauen jeweils den Baum auf.

Beispiel für LZ 78:
Der String `ababcbababaaaaaaa` erzeugt den folgenden Baum:



Input	a	b	ab	c	ba	bab	aa	aaa	aa
Output	0a	0b	1b	0c	2a	5b	1a	7a	7EOF

2.9 LZW-Komprimierung (Lempel/Ziv/Welch, 1984)

Starte mit Stringtabelle, gefüllt mit `characters`, und fülle sie mit Verweisen auf bereits gelesene Substrings.

```
x := get_char();
w := x;
repeat
   x := get_char();
   if wx in Tabelle
      then w := wx
      else put_string(code(w));
           trage wx in Tabelle ein
           w := x
   endif
until x = EOF
```

w	Input	Output	String	Code
			a	1
			b	2
			c	3
	a			
a	b	1	ab	4
b	a	2	ba	5
a	b			
ab	c	4	abc	6
c	b	3	cb	7
b	a			
ba	b	5	bab	8
b	a			
ba	b			
bab	a	8	baba	9
a	a	1	aa	10
a	a			
aa	a	10	aaa	11
a	a			
aa	a			
aaa	a	11	aaaa	12

Entwicklung von Output und Tabelle für den Input `ababcbababaaaaaaa`

String Präfix-String-Index	Erweiterungs-character	Code
	a	1
	b	2
	c	3
1	b	4
2	a	5
4	c	6
3	b	7
5	b	8
8	a	9
1	a	10
10	a	11
11	a	12

Implementation der Stringtabelle

LZW-Dekomprimierung (1. Näherung)

```
code := get_bits();
v := String(code);
put_char(v);
code := get_bits();
while code # EOF-code do
    w := String(code);
    put_string(w);
    fuege v * w[0] in Tabelle ein;
    v := w;
    code := get_bits()
end;
```

Obacht: Der Spezialfall eines Eingabestrings der Form `xwxwx` muß abgefangen und gesondert behandelt werden, da der Komprimierer unmittelbar nach Eintragen des Codes für `xwx` den Code auch sendet. Der Dekomprimierer hat ihn jedoch noch nicht eingetragen, kann es aber nachholen.

v	Input-Code	w	Tabellen-String	Tabellen-Code
			a	1
			b	2
			c	3
	1			
a	2	b	ab	4
b	4	ab	ba	5
ab	3	c	abc	6
c	5	ba	cb	7
ba	8	?		

Kompressionsraten für den Spiegelartikel *Multimedia*

```
                        spiegel.txt     16617  100 %
    Huffmann    pack    spiegel.txt.z    9944  60 %
    LZW         compress spiegel.txt.Z   9120  55 %
    Lempel-Ziv  gzip    spiegel.txt.gz   7765  47 %
```

Kapitel 3

Binärbild

Jeder Bildpunkt eines Binärbildes hat Wert 0 oder 1.
Binärbilder werden erzeugt

- durch Einscannen einer Schwarz/Weiß-Vorlage,
- durch Binärisierung eines Grauwertbildes,
- durch Algorithmen (z.B. zur Erzeugung einer Linie).

Zur Kompression eignet sich

- Run Length Encoding,
- LZW,
- CCITT T4 Komprimierung (Fax).

3.1 Kompression nach CCITT Gruppe T4

Für jede Zeile des Bildes werden zunächst die Längen der weißen und schwarzen Pixel-Sequenzen bestimmt und dann diese Folge positiver Zahlen mit einem Huffman-Code komprimiert.

CCITT-T4-Codes

Termination-Code Weiß

Länge	Code-Word
0	00110101
1	000111
2	0111
3	1000
4	1011
5	1100
6	1110
7	1111
8	10011
9	10100
10	00111
11	01000
12	001000
13	000011
14	110100
15	110101
16	101010
17	101011
18	0100111
19	0001100
20	0001000
21	0010111
22	0000011
23	0000100
24	0101000
25	0101011
26	0010011
27	0100100
28	0011000
29	00000010
30	00000011
31	00011010

Termination-Code Schwarz

Länge	Code-Word
0	0000110111
1	010
2	011
3	10
4	011
5	0011
6	0010
7	00011
8	000101
9	000100
10	0000100
11	0000101
12	0000111
13	00000100
14	00000111
15	000011000
16	0000010111
17	0000011000
18	0000001000
19	00001100111
20	00001101000
21	00001101100
22	00000110111
23	00000101000
24	00000010111
25	00000011000
26	000011001010
27	000011001011
28	000011001100
29	000011001101
30	000001101000
31	000001101001

Termination-Code Weiß

Länge	Code-Word
32	00011011
33	00010010
34	00010011
35	00010100
36	00010101
37	00010110
38	00010111
39	00101000
40	00101001
41	00101010
42	00101011
43	00101100
44	00101101
45	00000100
46	00000101
47	00001010
48	00001011
49	01010010
50	01010011
51	01010100
52	01010101
53	00100100
54	00100101
55	01011000
56	01011001
57	01011010
58	01011011
59	01001010
60	01001011
61	00110010
62	00110011
63	00110100

Termination-Code Schwarz

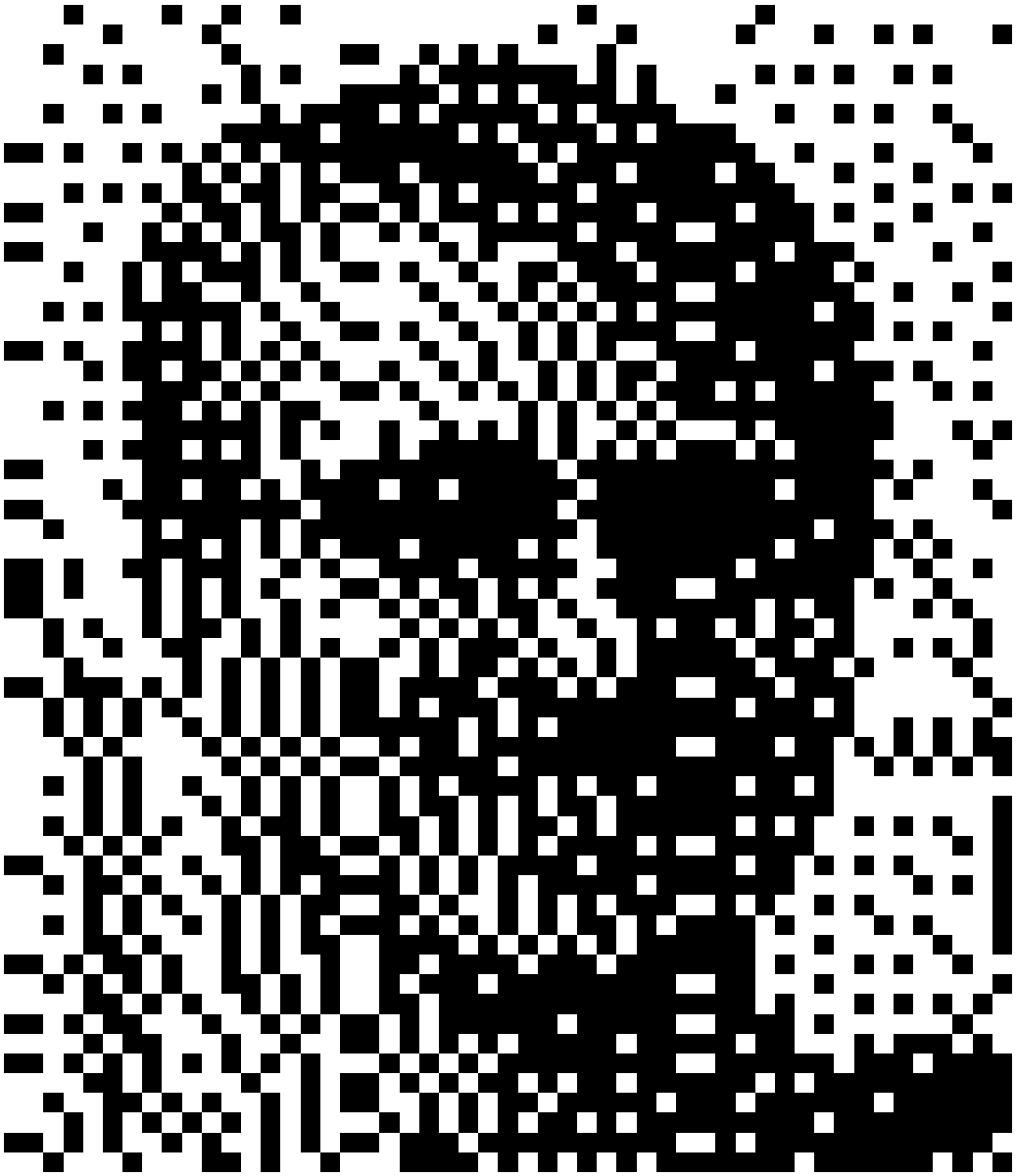
Länge	Code-Word
32	000001101010
33	000001101011
34	000011010010
35	000011010011
36	000011010100
37	000011010101
38	000011010110
39	000011010111
40	000001101100
41	000001101101
42	000011011010
43	000011011011
44	000001010100
45	000001010101
46	000001010110
47	000001010111
48	000001100100
49	000001100101
50	000001010010
51	000001010011
52	000000100100
53	000000110111
54	000000111000
55	000000100111
56	000000101000
57	000001011000
58	000001011001
59	000000101011
60	000000101100
61	000001011010
62	000001100110
63	000001100111

Makeup-Code Weiß

Länge	Code-Wort
64	11011
128	10010
192	010111
256	0110111
320	00110110
384	00110111
448	01100100
512	01100101
576	01101000
640	01100111
704	011001100
768	011001101
832	011010010
896	011010011
960	011010100
1024	011010101
1088	011010110
1152	011010111
1216	011011000
1280	011011001
1344	011011010
1408	011011011
1472	010011000
1536	010011001
1600	010011010
1664	011000
1728	010011011
EOL	000000000001

Makeup-Code Schwarz

Länge	Code-Word
64	0000001111
128	000011001000
192	000011001001
256	000001011011
320	000000110011
384	000000110100
448	000000110101
512	0000001101100
576	0000001101101
640	0000001001010
704	0000001001011
768	0000001001100
832	0000001001101
896	0000001110010
960	000000111011
1024	0000001110100
1088	0000001110101
1152	0000001110110
1216	0000001110111
1280	0000001010010
1344	0000001010011
1408	0000001010100
1472	0000001010101
1536	0000001011010
1600	0000001011011
1664	0000001100100
1728	0000001100101
EOL	000000000001



S/W-Bild mit 51 x 59 Pixeln

```
1110 1111 0110 1101 1111 1111 1111 1011 1111 1101 1111 1111 1110 0000
   e    f    6    d    f    f    f    b    f    d    f    f    e    0


 111   0 1111   0   11   0   11   0 11111111111111   0 11111111   0 111111111111              00000
   w   s    w   s    w   s    w   s              w   s        w   s            w         auffuellen


   3   1    4   1    2   1    2   1             14   1        8   1           12                EOL
1000 010 1011 010 0111 010 0111 010         110100 010    10011 010       001000       000000000001

0000 0000 0000 0001 1100 0010 1011 0100 1110 1001 1101 0110 1000 1010 0110 1000 1000 0000 0000 0001
   0    0    0    1    c    2    b    4    e    9    d    6    8    a    6    8    8    0    0    1
|---> Vorspann <----|
```

Erste Zeile eines Schwarz/Weiß-Bildes mit 51 x 59 Pixeln
unkomprimiert, Weiß wird mit 1 codiert, Schwarz wird mit 0 codiert, je 8 Bits pro Byte
komprimiert nach CCITT Gruppe 4

```
4d4d 002a 0000 01a6

ef 6d ff fb fd ff e0 fb df ff ee fb b5 c0
df ef 9a bd ff ff e0 f5 f5 f4 05 7d 5a e0
ff d7 82 a1 77 ff e0 da fa 14 14 3e d6 e0
ff e0 81 42 87 ff 60 2d 54 00 28 03 77 a0
ff aa 88 11 09 dd e0 ea 92 65 04 00 f7 40
3f 4a 94 50 84 5d e0 f6 a9 6b 04 30 77 a0
3e 15 7d 71 01 1e e0 ed 45 96 c8 84 2f c0
fd 36 fb 54 30 1b 60 d4 0b 7d aa 80 4f c0
fe ad 96 d4 30 0a e0 2c 2a fb 55 84 1f a0
f4 95 6d 6a 40 45 e0 fe 2b 96 aa 8a 0e a0
d4 54 fb d5 40 07 e0 fe 05 6d 56 ba 07 40
f4 55 e8 15 45 0f a0 3e 06 80 08 00 05 e0
fa 4a 12 04 01 0b a0 3c 01 00 08 00 0f c0
de 8a 00 04 00 45 e0 fe 2a 88 2c 01 0a e0
2c 8d 61 06 04 0d a0 2e ab 94 ac 30 16 e0
3e ad 6a 96 02 9d 60 d6 95 f5 5b 4a 5f a0
db 35 6a 2d 04 9a a0 ef aa 9a 55 02 2f 60
22 aa 90 8a 31 1f a0 d5 6a 94 40 04 5f c0
d5 aa 81 50 00 1a a0 eb d5 69 00 31 2a 80
f5 ea 90 40 00 35 40 d5 b5 6a 14 84 bf e0
f5 d5 69 52 00 3f c0 d5 69 65 4a 05 6a c0
ea e4 95 50 b0 7f 40 35 b4 6a 84 04 ab c0
d2 d5 0a a0 80 fd 40 f5 6a 65 aa 35 af c0
d5 aa 8a a9 42 f5 c0 f2 ea 61 55 03 be c0
29 6b 64 22 02 eb 60 d5 65 68 80 33 bf c0
f2 b5 64 00 02 ea a0 29 da 94 08 30 bf 60
f4 ed 95 41 04 e8 a0 2a aa ea 80 30 22 00
f2 f6 95 29 02 80 00 d9 5a 9a 90 44 08 00
ea aa ea a5 00 40 00 2b 5a 91 00 34 00 20
dd cb 70 aa 80 81 40 00

000b

0100 0003 0000 0001 0033 0000 0101 0003 0000 0001 003b 0000
0102 0003 0000 0001 0001 0000 0103 0003 0000 0001 0001 0000
0106 0003 0000 0001 0001 0000 0111 0004 0000 0001 0000 0008
0112 0003 0000 0001 0001 0000 0115 0003 0000 0001 0001 0000
0116 0004 0000 0001 0000 003b 0117 0004 0000 0001 0000 019d
011c 0003 0000 0001 0001 0000

0000 0000
```

Hex-Dump einer 51 x 59 S/W-TIF-Datei, unkomprimiert

```
4d4d 002a 0000 03ee

0001 c2b4 e9d6 8a68 8001 0820 4290 8410 91d2 4611 7447 4a00 01ba 9acd d0e8 7568
4000 0109 1d11 d328 72b1 2107 91d1 c447 4474 5d11 d400 019d 0eac c743 a1d8 ea16
8001 1745 d020 42ca 1c10 2485 91d0 42c4 ba2e 88e8 ba80 01a0 c721 d0ec 743b 4270
0113 5c8e 8ba0 450e 50e0 8108 b23a 0845 9746 1350 01d1 0e87 43a1 e1c8 7876 3f0a
1580 0118 4474 1056 b613 23a2 3984 22c4 2084 104a 0001 9afc 87c7 43a1 f1d0 f0e8
763b 1e1d 42b0 0109 7492 082d a092 4082 1104 0841 0431 0821 0940 019a f8c7 43a1
d621 d443 8c7a d400 0109 1d22 8708 21ec e384 d040 8132 3b08 511d 0a80 01f2 1f74
eb13 a1d0 e876 e374 e9c0 0117 4105 10c2 0995 8134 9228 7081 0840 988a 8001 fa1d
0e9d 3e3a 74e8 743b 7147 43a8 0001 135c 8e90 f482 042c 2617 c102 1287 4221 2800
01da 1f1f 1d0e 874e 9d0e 9d0e 87c7 18f0 eac0 010b 1492 4d04 1341 05f4 8c38 2042
9820 9400 01ba 1d0f 0e87 43f8 ad0e 8743 a1c2 f000 0108 b11d 11d0 4132 870b da45
4143 c104 a001 da1e 1d0e 8756 8721 d0e8 743c 3a1d e21c 0001 135c 588d 9438 b422
22c1 0218 0001 e21f 1f1d 0ec7 c721 c41d 8ea1 0e00 0113 5d88 b238 7431 7001 bab4
3c3a 1c1c 70a3 c3ab 0001 0c42 0bca 1cc3 8204 a104 2922 3a80 019a e3a7 c79d 0e9d
8e6e 4393 a743 8001 7bc2 0549 9c70 8264 7447 4810 4282 0984 3001 9af8 8743 a743
a743 a1d0 f8e9 c63a 1f84 3a70 0106 50e0 bb08 2c5a 4904 1594 e50e 105b 23a1 6a00
01ba 74fb a1d0 e9d0 e878 74e8 731f 1f74 3a1d 0e00 0130 85d2 4920 9749 0417 610a
0821 6a00 019a e3c3 a1d0 e874 3a1f 1d8f 0e87 9e1e e438 0001 0608 1249 2ff8 204c
1088 8410 8447 4e00 01ba 1d0e 9d0e 8743 a1c8 743a 1c30 e874 3a1c 0001 3493 4924
8a70 40b1 124e 61d0 417d 8001 da1d 5a1d 0e87 c731 c177 43a1 d0e8 0001 0417 8413
4920 9143 8204 250e 50e7 1cb1 c205 1116 0001 da1d 421d 0e87 4e87 c743 a1f1 c2cd
0001 0417 8413 4bd8 5ec2 c90e 088e 8ba2 3a04 0b00 01c2 1d0e a31f 1f1d 0e87 43a1
d8e9 cfd0 e800 0102 09a4 8204 f692 4921 0408 44a1 d11d 11d0 2043 8001 ba1f 1d3a
1d0e 8763 a1d0 e873 1c7c 8743 a001 0d24 104d 2481 04d2 e50e 1042 5382 23a2 e810
211c 0001 ba1d 0e9d 0e87 43a1 e1d0 e874 3a1f 1d0e c756 8750 8001 0d84 10fd 842d
2490 4163 1040 8582 c001 9ae3 a1f1 d3a1 d3a7 c763 c38c 7508 74e9 c001 0690 417b
082e 9041 0899 e210 4222 a001 de3a 1d3a 1d0e 9f1c 171d 421d 0e87 4380 0113 5d04
1361 3490 4105 921c 9b8c 2084 5a80 01da 1f84 e9f1 d0e8 7439 8e63 f087 8743 8001
135d 2411 1d04 1690 4083 4924 8410 421c 4208 5800 01de 3ab4 e87c 743a 1f1d 0f8e
43a1 c200 0104 09b2 3a08 2617 b482 09a1 2874 84c3 c001 c21d 0e87 43a1 d0ea 10e8
743a 1f1d 0e20 e200 0102 0bda 4bc2 49a1 113b 9438 a800 01ba 8518 e9d4 31d0 e874
3a1c 6390 e874

000c
0100 0003 0000 0001 0033 0000 0101 0003 0000 0001 003b 0000
0102 0003 0000 0001 0001 0000 0103 0003 0000 0001 0003 0000
0106 0003 0000 0001 0001 0000 0111 0004 0000 0001 0000 0008
0112 0003 0000 0001 0001 0000 0115 0003 0000 0001 0001 0000
0116 0004 0000 0001 0000 003b 0117 0004 0000 0001 0000 03e6
011c 0003 0000 0001 0001 0000 0124 0004 0000 0001 0000 0005

0000 0000
```

Hex-Dump einer 51 x 59 S/W-TIF-Datei, komprimiert nach CCITT Gruppe 4

```
4d4d 002a 0000 01e0

06ef 6dff fbfd ffe0 06fb dfff eefb b5c0
06df ef9a bdff ffe0 fff5 04f4 057d 5ae0
06ff d782 a177 ffe0 06da fa14 143e d6e0
06ff e081 4287 ff60 062d 5400 2803 77a0
06ff aa88 1109 dde0 06ea 9265 0400 f740
063f 4a94 5084 5de0 06f6 a96b 0430 77a0
063e 157d 7101 1ee0 06ed 4596 c884 2fc0
06fd 36fb 5430 1b60 06d4 0b7d aa80 4fc0
06fe ad96 d430 0ae0 062c 2afb 5584 1fa0
06f4 956d 6a40 45e0 06fe 2b96 aa8a 0ea0
06d4 54fb d540 07e0 06fe 056d 56ba 0740
06f4 55e8 1545 0fa0 063e 0680 0800 05e0
06fa 4a12 0401 0ba0 063c 0100 0800 0fc0
06de 8a00 0400 45e0 06fe 2a88 2c01 0ae0
062c 8d61 0604 0da0 062e ab94 ac30 16e0
063e ad6a 9602 9d60 06d6 95f5 5b4a 5fa0
06db 356a 2d04 9aa0 06ef aa9a 5502 2f60
0622 aa90 8a31 1fa0 06d5 6a94 4004 5fc0
06d5 aa81 5000 1aa0 06eb d569 0031 2a80
06f5 ea90 4000 3540 06d5 b56a 1484 bfe0
06f5 d569 5200 3fc0 06d5 6965 4a05 6ac0
06ea e495 50b0 7f40 0635 b46a 8404 abc0
06d2 d50a a080 fd40 06f5 6a65 aa35 afc0
06d5 aa8a a942 f5c0 06f2 ea61 5503 bec0
0629 6b64 2202 eb60 06d5 6568 8033 bfc0
06f2 b564 0002 eaa0 0629 da94 0830 bf60
06f4 ed95 4104 e8a0 062a aaea 8030 2200
06f2 f695 2902 8000 06d9 5a9a 9044 0800
06ea aaea a500 4000 062b 5a91 0034 0020
06dd cb70 aa80 8140

000b

0100 0003 0000 0001 0033 0000 0101 0003 0000 0001 003b 0000
0102 0003 0000 0001 0001 0000 0103 0003 0000 0001 8005 0000
0106 0003 0000 0001 0001 0000 0111 0004 0000 0001 0000 0008
0112 0003 0000 0001 0001 0000 0115 0003 0000 0001 0001 0000
0116 0004 0000 0001 0000 003b 0117 0004 0000 0001 0000 01d8
011c 0003 0000 0001 0001 0000

0000 0000
```

Hex-Dump einer 51 x 59 S/W-TIF-Datei, komprimiert nach Run Length Encoding
(keine Einsparung, da zu wenig identische Zeichen)

3.2 Scanner & OCR

Ein Flachbettscanner tastet eine Schwarz/Weiß-Vorlage, z.B. einen Brief mit schwarzer Schrift auf weißem Grund, zeilenweise ab und erstellt ein Binärbild. Bei einer Auflösung von 400 dpi ergibt eine Briefmarke etwa 200.000 Pixel, eine DIN-A4-Seite etwa 15 Millionen Pixel. Diese Datei kann (ggf. komprimiert) als Fax verschickt werden.

Per OCR-Software (*Optical Character Recognition*) kann das empfangene Bild analysiert werden und, mit einer gewissen Fehlerrate, in ASCII-Text konvertiert werden.

Ursprünglich gab es spezielle Scannerfonts, dann wurden für die gängigen Fonts die zugehörigen Bitmaps hinterlegt, heutzutage werden die gelesenen Buchstaben skelettiert und in ihre grafischen Grundelemente zerlegt (Omnifont-Technologie). Die Verarbeitungsgeschwindigkeit steigt mit der Zahl der eingescannten Seiten. Ggf. werden ein allgemeines Lexikon, benutzerdefiniertes Lexikon und Sprachregeln zu Rate gezogen. (Eine Sprachregel verbietet z.B. einzelne Großbuchstaben innerhalb von Wörtern.) Bei Unterschreiten eines vorgewählten Sicherheitsfaktors wird die fragliche Textstelle markiert. Durch interaktive Erstellung eines Trainingssatzes kann dem System das Erkennen unklarer Pattern erleichtert werden. Bei mehrspaltigem Satz kann der Benutzer die Textbereiche identifizieren und einordnen.

```
\section{Ein FEM-Verfahren}
\subsection{Methode der finiten Elemente}

FEM-Verfahren geh"oren zu den weitverbreitetsten numerischen L"osungsverfahren
f"ur Differentialgleichungen auf endlichen Gebieten. Das Berechnungsgebiet
wird dabei durch ein Netz von Zwischenpunkten in kleine Teilgebiete
zerlegt --- die finiten Elemente. Die L"osung der Differentialgleichung wird
jetzt elementweise approximiert, indem "uber jedem Element die Integralform
der DGl numerisch gel"ost wird. Dabei ist es n"otig, da"s der Wert
der L"osungsfunktion in den Netzpunkten bekannt ist. Da dies nicht der
Fall ist, wird hierf"ur ein angen"aherter Wert benutzt, der auf die gleiche
Weise berechnet wird. Es ergibt sich also eine Folge von N"aherungen in jedem
Netzpunkt. W"ahlt man eine geeignete Startl"osung, so kann bewiesen werden,
da"s diese Folge gegen die exakte L"osung konvergiert, falls die Netzdichte
klein genug ist. F"ur Punkte zwischen den Netzpunkten wird die L"osung
mit Hilfe des zugeh"origen Elements interpoliert.

F"ur die folgenden Betrachtungen gehen wir von einer einzelnen Funktion $f$
auf einem zweidimensionalen Gebiet aus. Diese Funktion wird mit Hilfe
sogenannter Ansatzfunktionen $N_i$ und der Funktionswerte an den Knoten $f_i$
approximiert:
\begin{equation}
  f(x, y) = \sum_{i=1}^3 N_i f_i .
\end{equation}

Diese Ansatzfunktionen h"angen im wesentlichen von der Geometrie der finiten
Elemente ab. Wir verwenden bei zweidimensionalen Problemen nur lineare
Dreieckselemente, auf denen wir zur Bestimmung der $N_i$ ein spezielles
Koordinatensystem einf"uhren. Betrachten wir dazu ein solches Dreieck mit
den Koordinaten $(x_1, y_1), (x_2, y_2)$ und $(x_3, y_3)$ (Bild~2.1).

Wir f"uhren sogenannte Fl"achenkoordinaten $L_1, L_2$ und $L_3$ ein mit:

\begin{eqnarray}
x & = & L_1 x_1 + L_2 x_2 + L_3 x_3 \nonumber \\
y & = & L_1 y_1 + L_2 y_2 + L_3 y_3 \\
1 & = & L_1 + L_2 + L_3 . \nonumber
\end{eqnarray}
```

Datei im Latex-Format

2.2 Ein FEM-Verfahren

2.2.1 Methode der finiten Elemente

FEM-Verfahren gehören zu den weitverbreitetsten numerischen Lösungsverfahren für Differentialgleichungen auf endlichen Gebieten. Das Berechnungsgebiet wird dabei durch ein Netz von Zwischenpunkten in kleine Teilgebiete zerlegt — die finiten Elemente. Die Lösung der Differentialgleichung wird jetzt elementweise approximiert, indem über jedem Element die Integralform der DGl numerisch gelöst wird. Dabei ist es nötig, daß der Wert der Lösungsfunktion in den Netzpunkten bekannt ist. Da dies nicht der Fall ist, wird hierfür ein angenäherter Wert benutzt, der auf die gleiche Weise berechnet wird. Es ergibt sich also eine Folge von Näherungen in jedem Netzpunkt. Wählt man eine geeignete Startlösung, so kann bewiesen werden, daß diese Folge gegen die exakte Lösung konvergiert, falls die Netzdichte klein genug ist. Für Punkte zwischen den Netzpunkten wird die Lösung mit Hilfe des zugehörigen Elements interpoliert.

Für die folgenden Betrachtungen gehen wir von einer einzelnen Funktion f auf einem zweidimensionalen Gebiet aus. Diese Funktion wird mit Hilfe sogenannter Ansatzfunktionen N_i und der Funktionswerte an den Knoten f_i approximiert:

$$f(x,y) = \sum_{i=1}^{3} N_i f_i. \tag{2.1}$$

Diese Ansatzfunktionen hängen im wesentlichen von der Geometrie der finiten Elemente ab. Wir verwenden bei zweidimensionalen Problemen nur lineare Dreieckselemente, auf denen wir zur Bestimmung der N_i ein spezielles Koordinatensystem einführen. Betrachten wir dazu ein solches Dreieck mit den Koordinaten $(x_1, y_1), (x_2, y_2)$ und (x_3, y_3) (Bild 2.1).

Wir führen sogenannte Flächenkoordinaten L_1, L_2 und L_3 ein mit:

$$\begin{aligned} x &= L_1 x_1 + L_2 x_2 + L_3 x_3 \\ y &= L_1 y_1 + L_2 y_2 + L_3 y_3 \\ 1 &= L_1 + L_2 + L_3. \end{aligned} \tag{2.2}$$

von der Latex-Quelle generierter Ausdruck

2.2 Ein FEM-Verfahren

2.2.1 Methode der finiten Elemente

FEM-Verfahren gehören zu den weitverbreitetsten numerischen Lösungsverfahren für Differentialgleichungen auf endlichen Gebieten. Das Berechnungsgebiet wird dabei durch ein Netz von Zwischenpunkten in kleine Teilgebiete zerlegt — die finiten Elemente. Die Lösung der Differentialgleichung wird jetzt elementweise approximiert, indem über jedem Element die Integralform der DGl numerisch gelöst wird. Dabei ist es nötig, daß der Wert der Lösungsfunktion in den Netzpunkten bekannt ist. Da dies nicht der Fall ist, wird hierfür ein angenäherter Wert benutzt, der auf die gleiche Weise berechnet wird. Es ergibt sich also eine Folge von Näherungen in jedem Netzpunkt. Wählt man eine geeignete Startlösung, so kann bewiesen werden, daß diese Folge gegen die exakte Lösung konvergiert, falls die Netzdichte klein genug ist. Für Punkte zwischen den Netzpunkten wird die Lösung mit Hilfe des zugehörigen Elements interpoliert.

Für die folgenden Betrachtungen gehen wir von einer einzelnen Funktion f auf einem zweidimensionalen Gebiet aus. Diese Funktion wird mit Hilfe sogenannter Ansatzfunktionen N_i und der Funktionswerte an den Knoten f_i approximiert:

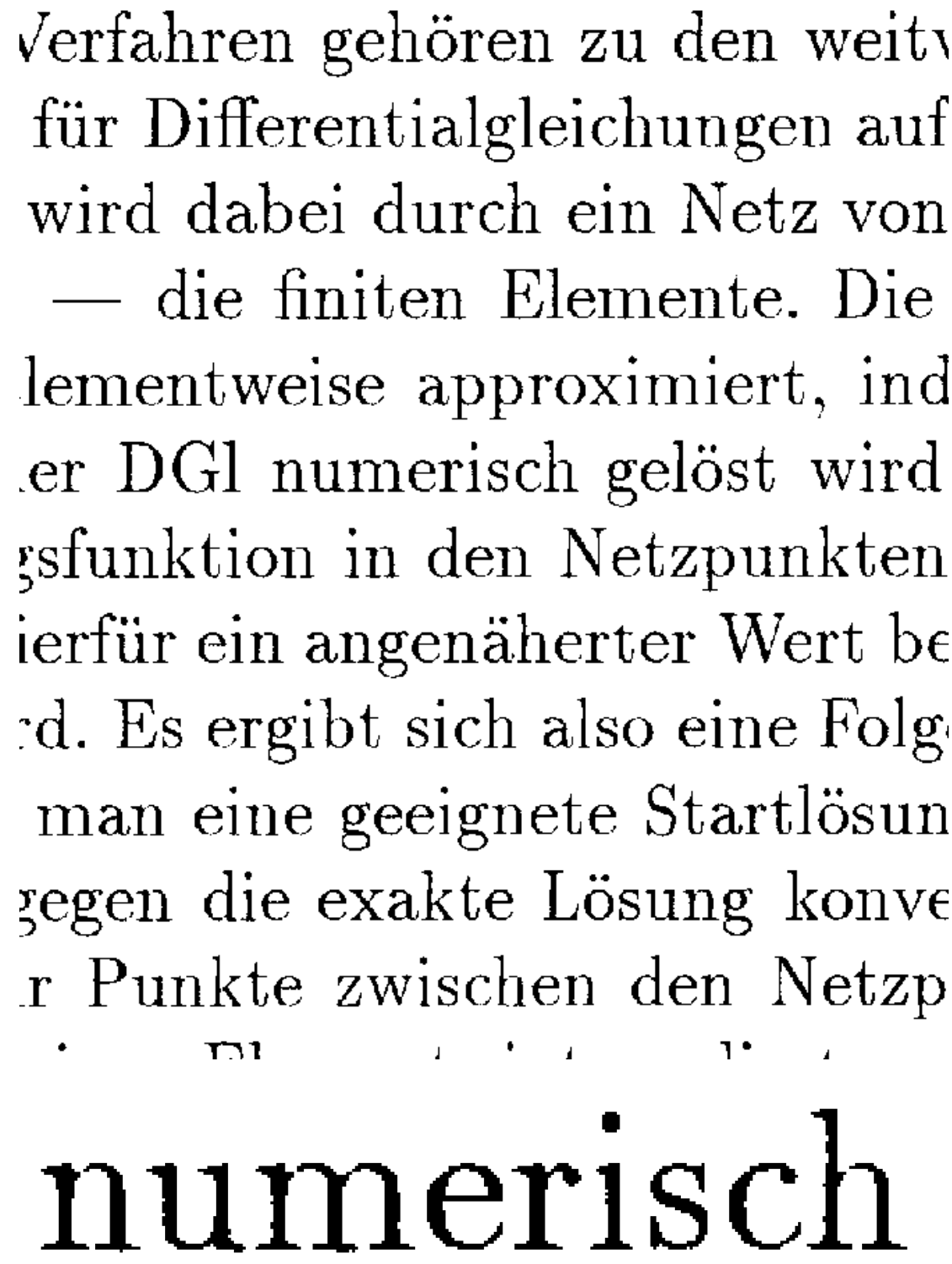
$$f(x,y) = \sum_{i=1}^{3} N_i f_i. \tag{2.1}$$

Diese Ansatzfunktionen hängen im wesentlichen von der Geometrie der finiten Elemente ab. Wir verwenden bei zweidimensionalen Problemen nur lineare Dreieckselemente, auf denen wir zur Bestimmung der N_i ein spezielles Koordinatensystem einführen. Betrachten wir dazu ein solches Dreieck mit den Koordinaten $(x_1, y_1), (x_2, y_2)$ und (x_3, y_3) (Bild 2.1).

Wir führen sogenannte Flächenkoordinaten L_1, L_2 und L_3 ein mit:

$$\begin{aligned} x &= L_1 x_1 + L_2 x_2 + L_3 x_3 \\ y &= L_1 y_1 + L_2 y_2 + L_3 y_3 \\ 1 &= L_1 + L_2 + L_3. \end{aligned} \tag{2.2}$$

Ergebnis des Scanners



Zwei Ausschnittvergrößerungen der eingescannten Seite

```
{\rtf0\pc{\fonttbl{\f0\fswiss Chicago;}
{\f2\froman New York;}{\f3\fswiss Geneva;}
{\f4\fmodern Monaco;}{\f5\fscript Venice;}
{\f6\fdecor London;}{\f7\fdecor Athens;}
{\f8\fdecor San Francisco;}
{\f11\fnil Cairo;}
{\f12\fnil Los Angeles;}
{\f20\froman Times;}
{\f21\fswiss Helvetica;}
{\f22\fmodern Courier;}{\f24\fnil Mobile;}}
{\stylesheet
{\s3\noline\plain \f20\fs36 \qj\sl000\li940\fi-940\tx940\f20  para3;}
{\s4\noline\plain \f20\fs36 \b\qj\sl000\li1040\fi-1040\tx1040\f20  para4;}
{\s5\noline\plain \f20\fs24 \qj\sl000\li0\fi0\f20  para5;}
{\s7\noline\plain \f20\fs24 \qc\sl000\li0\fi0\f20  cent7;}
{\s6\noline\plain \f20\fs24 \qj\sl000\li0\fi360\f20  para6;}
{\s8\noline\plain \f20\fs20 \qc\sl000\li0\fi0\f20  cent8;}
{\s1\noline\plain \f20\fs24 \ql\sl000\li0\fi0\ri0\keep\tx3000\tqdec\tx7760\f20  table1;}
{\s2\noline\plain \f20\fs20 \ql\sl000\li0\fi0\ri0\keep\tx2520\tx2880\tx3240\tqdec\tx7740\f20  table2;}
{\s3\noline\plain \f20\fs36 \qj\sl000\li940\fi-940\tx940\f20  para3;}
{\s4\noline\plain \f20\fs36 \b\qj\sl000\li1040\fi-1040\tx1040\f20  para4;}
{\s5\noline\plain \f20\fs24 \qj\sl000\li0\fi0\f20  para5;}
{\s7\noline\plain \f20\fs24 \qc\sl000\li0\fi0\f20  cent7;}
{\s6\noline\plain \f20\fs24 \qj\sl000\li0\fi360\f20  para6;}
{\s8\noline\plain \f20\fs20 \qc\sl000\li0\fi0\f20  cent8;}}
\sectd \cols1\endnhere \pard\plain \s3\noline\plain \f20\fs36 \qj\sl000\li940\fi-940\tx940\f20
{\plain \f20\fs36 2.2\tab Ein F‘EM-Verfahren}
\par \par \pard\plain \s4\noline\plain \f20\fs36 \b\qj\sl000\li1040\fi-1040\tx1040\f20
{\plain \f20\fs36 \b 2.2.1\tab Methode der finiten Elemente}
\par \pard\plain \s5\noline\plain \f20\fs24 \qj\sl000\li0\fi0\f20
{\plain \f20\fs24 FEM-Verfahren gehren zu d
en weitverbreitet sten nun˜erischen Lsungsverfahren fr D ifferent i alglei d˜ung
en auf endli den Gebieten. Das  erechnungsgebiet wird dabei durch ein Netz von
Zwisd˜enpunkten in kleine Teilgebiete zerlegt die finiten Elen˜ente. Die Lsung der
 Differentialgleid˜ung wird jetzt eIen˜entweise approxiu˜iert, inden˜ ber jeden- Ele
n˜ent die Integralforn˜ der DGI nui˜erisd˜ gelst wird. Dabei ist es ntig, dat der
Wert der Lsungsfunktion in den Netzpunkten bekannt ist. Da dies nicht der Fall i
st, wird hierfr ein angenherter Wert benutzt, der auf die gleid˜e Weise berechue
t wird. Bs ergibt sid˜ also eine Folge von Nherungen in jeden- Netzpunkt. Whlt i
 exakte Lsung konvergiert, falls die Netzdid˜te klein genug ist. Fr Punkte zwisc
hen den Netzpunkten wird die Lsung i˜it liilfe des zugehrigen Elen˜ents interpoli
ert.}\par \pard\plain \s6\noline\plain \f20\fs24 \qj\sl000\li0\fi360\f20
{\plain \f20\fs24 Fr die folgenden Betradftungen gehen wir von einer einzelnen Funktion f auf
einen- zweidin˜ensionalen Gebiet ans. Diese Funktion wird n˜it llilfe sogenannter A
nsatzfunktionen N˜ und der Funktionswerte an den Nnoten ft approxin˜iert:}
\par \pard\plain \s7\noline\plain \f20\fs24 \qc\sl000\li0\fi0\f20
{\plain \f20\fs24 3}\par \pard\plain \s1\noline\plain \f20\fs24 \ql\sl000\li0\fi0\ri0\keep\tx3000\tqdec\tx7760\f20
\tab {\plain \f20\fs24 f(x,y) }{\plain \f20\fs20 =\tab }{\plain \f20\fs24 (
2.1)}\par \pard\plain \s7\noline\plain \f20\fs24 \qc\sl000\li0\fi0\f20
{\plain \f20\fs24 i=I}\par \par \pard\plain \s6\noline\plain \f20\fs24 \qj\sl000\li0\fi360\f20
{\plain \f20\fs24 Diese Ansatzfunktionen hngen ins wesentlichen von der Geon˜etrie der finit
en Elen˜ente ab. Wir verwenden bei zweidin˜ensionalen Problen˜en nur lineare Dreied˜s
elen˜ente, auf denen wir zur Bestin˜n˜ung der N˜ ein spezielles l˜oordinatensysten˜ ein
fhren. Betradften wir dazu ein sold˜es Dreieck n˜it den Noordinaten (x}{\dn6 1}{, yi), (x}{\dn6 2}{,
 Y2) und (x}{\dn6 3}{, y˜) (Bild 2.1).}\par \pard\plain \s6\noline\plain \f20\fs24 \qj\sl000\li0\fi360\f20
{\plain \f20\fs24 Wir fhren sogenannte Flchenkoordinaten L}{\dn6 1}{, L}{\dn6 2 }{und L}{\dn6 3 }{ein n˜it:}
\par \par \par \pard\plain \s2\noline\plain \f20\fs20 \ql\sl000\li0\fi0\ri0
\keep\tx2520\tx2880\tx3240\tqdec\tx7740\f20 \tab {\plain \f20\fs24 x\tab }
{\plain \f20\fs20 =\tab L}{\dn6 1}{x}{\dn6 1 }{+ L}{\dn6 2}{x}{\dn6 2 }{+ L}{\dn6 3}{x}{\dn6 3}
{\line \tab y\tab =\tab L}{\dn6 1}{y}{\dn6 1 }{+ L}{\dn6 2}{y}{\dn6 2 }{+ L}{\dn6 3}{y}{\dn6 3\tab }
{(2.2)\line \tab \tab =\tab L}{\dn6 1}{+L}{\dn6 2}{+L}{\dn6 3}{.}
\par \par \par \par \par \pard\plain \s8\noline\plain \f20\fs20 \qc\sl000\li0\fi0\f20
{\plain \f20\fs20 42}}
```

von der OCR-Software erzeugte RTF-Datei (im Rich Text Format)

2.2 Ein F`EM-Verfahren

2.2.1 Methode der finiten Elemente

FEM-Verfahren gehören zu den weitverbreitet sten nun~erischen Lösungsverfahren für D ifferent i alglei d~ungen auf endli den Gebieten. Das ß erechnungsgebiet wird dabei durch ein Netz von Zwisd~enpunkten in kleine Teilgebiete zerlegt die finiten Elen~ente. Die Lösung der Differentialgleid~ung wird jetzt eIen~entweise approxiu~iert, inden~ über jeden- Elen~ent die Integralforn~ der DGI nui~erisd~ gelöst wird. Dabei ist es nötig, dat der Wert der Lösungsfunktion in den Netzpunkten bekannt ist. Da dies nicht der Fall ist, wird hierfür ein angenäherter Wert benutzt, der auf die gleid~e Weise berechuet wird. Bs ergibt sid~ also eine Folge von Näherungen in jeden- Netzpunkt. Wählt i~an eine geeignete Startlösung, so kann bewiesen werden, daf diese Folge gegen die exakte Lösung konvergiert, falls die Netzdid~te klein genug ist. Für Punkte zwischen den Netzpunkten wird die Lösung i~it liilfe des zugehörigen Elen~ents interpoliert.

Für die folgenden Betradftungen gehen wir von einer einzelnen Funktion f auf einen- zweidin~ensionalen Gebiet ans. Diese Funktion wird n~it llilfe sogenannter Ansatzfunktionen N~ und der Funktionswerte an den Nnoten ft approxin~iert:

$$f(x,y) = \sum_{i=I}^{3} \qquad (2.1)$$

Diese Ansatzfunktionen hängen ins wesentlichen von der Geon~etrie der finiten Elen~ente ab. Wir verwenden bei zweidin~ensionalen Problen~en nur lineare Dreied~selen~ente, auf denen wir zur Bestin~n~ung der N~ ein spezielles l~oordinatensysten~ einführen. Betradften wir dazu ein sold~es Dreieck n~it den Noordinaten (x_1, yi), (x_2, Y2) und (x_3, y~) (Bild 2.1).

Wir führen sogenannte Flächenkoordinaten L_1, L_2 und L_3 ein n~it:

$$\begin{aligned} x &= L_1x_1 + L_2x_2 + L_3x_3 \\ y &= L_1y_1 + L_2y_2 + L_3y_3 \\ &= L_1+L_2+L_3. \end{aligned} \qquad (2.2)$$

von der Rich-Text-Format-Quelle generierter Ausdruck

korrespondierende Ausschnitte

```
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
```

aus dem Hex-Dump zu `diss.tif` (keine Kompression)

```
81 ff
```

aus dem Hex-Dump zu `diss.tifll` (Run Length Encoding)
($81 `ff` = 257-129 mal Weiß)

Speicherbedarf der am Einscannen beteiligten Dateien:

File	Bytes	%	Inhalt
diss.tex	2.073	0.1	Quelle im tex-Format
diss.ps	32.657	1.8	durch Quelle erzeugte Postscript-Datei
diss.tif	1.947.078	100.0	vom Scanner erzeugte TIF-Datei (400 dpi => 3328 x 4680 Pixel)
diss.tifll	306.924	15.8	komprimiert mit Run Length Encoding
diss.tif5	153.126	7.8	komprimiert mit LZW
diss.tif3	127.932	6.5	komprimiert mit CCITT Group T4
diss.rtf	4.481	0.2	von OCR-Software erzeugte RTF-Datei

3.3 Binärisierung

Erzeugung eines Binärbildes $b[i,j]$ aus einem Grauwertbild $g[i,j]$:

Konstanter Schwellwert T

$$b[i,j] = \begin{cases} 1, \text{ falls } g[i,j] \geq T \\ 0 \text{ sonst} \end{cases}$$

Angepaßter Schwellwert
setze Schwelle

$$T := (\min_{i,j}\{g[i,j]\} + \max_{i,j}\{g[i,j]\})/2$$

Schwellwertverfahren eignen sich nur bei Vorlagen mit hohem Kontrast.

Error-Diffusion
Sei $g[i,j] \in \{0,\ldots,255\}$, sei $P(i,j) = 255 \cdot b[i,j]$
Verteile den Fehler $E[i,j] = g[i,j] - P[i,j]$ auf die Nachbarpunkte:

$$\begin{aligned}
g[i+1,j] &= g[i+1,j] &+ \tfrac{7}{16}E[i,j] \\
g[i+1,j+1] &= g[i+1,j+1] &+ \tfrac{1}{16}E[i,j] \\
g[i,j+1] &= g[i,j+1] &+ \tfrac{5}{16}E[i,j] \\
g[i-1,j+1] &= g[i-1,j+1] &+ \tfrac{3}{16}E[i,j]
\end{aligned}$$

		7
3	5	1

Pattern-Dither
Ändere den Schwellwert von Bildpunkt zu Bildpunkt. Eine $N \times N$ Dithermatrix T ist mit den N^2 Zahlen von 0 bis $N^2 - 1$ besetzt.

$$b[i,j] = \begin{cases} 1, \text{ falls } g[i,j] \geq T[i \mod n, j \mod n] \\ 0 \text{ sonst} \end{cases}$$

Die Einträge müssen so verteilt sein, daß sich Quadrate gleichen Grauwerts gut aneinanderfügen lassen.

Error-Diffusion erzeugt *Echos* und *Geisterbilder*. *Ordered-Dither* vergröbert; Strukturen kleiner als N können verschwinden.

Diffusion-Dither
Über eine $N \times N$ Matrix C erhält jeder Bildpunkt i, j die Klasse $C[i \bmod N, j \bmod N]$.
Die Pixel werden nicht zeilenweise abgearbeitet, sondern klassenweise. Anhand eines konstanten Schwellwertes wird entschieden, ob $b[i,j]$ auf 0 oder 1 gesetzt werden soll. Der hierbei auftretende Fehler wird auf jene Nachbarn des Punktes verteilt, die einer höheren Klasse angehören.
Die Einträge müssen so verteilt sein, daß jede Zahl mindestens einen größeren Nachbarn hat.

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	52	20	62	30	54	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	55	23	61	29	53	21

8×8-Dithermatrix

25	21	13	39	47	57	53	45
48	32	29	43	55	63	61	56
40	30	35	51	59	62	60	52
36	14	22	26	46	54	58	44
16	6	10	18	38	42	50	24
8	0	2	7	15	31	34	20
4	1	3	11	23	33	28	12
17	9	5	19	27	49	41	37

8×8-Dot-Diffusion-Matrix

86 Graustufen

50 % Threshold

Pattern Dither

Diffusion Dither

Original und binärisierte Versionen eines Grauwertbildes im Format 108 x 157.

Kapitel 4

Grauwertbilder

Jeder Bildpunkt eines Grauwertbildes trägt einen Helligkeitswert, typischerweise codiert als Byte im Bereich 0 bis 255. Bei Scannern und Bildschirmen ist die Beziehung zwischen Beschleunigungsspannung V der Elektronen und der durch sie erzeugten Lichtintensität I nicht linear, sondern über einen Exponenten γ (oft 0.4) gekoppelt:

$$I = k \cdot V^{\gamma}$$

Daher muß bei der Bilderzeugung die zur Intensität I passende Spannung V wie folgt bestimmt werden (γ-Korrektur):

$$V = \left(\frac{I}{k}\right)^{\frac{1}{\gamma}}$$

4.1 Geometrische Transformationen

Ein einzelner Punkt mit den Koordinaten x, y wird bewegt an die Stelle x', y':

Translation	$x' := x + k_1$
	$y' := y + k_2$
Skalierung bzgl. Nullpunkt	$x' := x \cdot k_1$
	$y' := y \cdot k_2$
Rotation bzgl. Nullpunkt	$x' := x \cdot \cos(\alpha) - y \cdot \sin(\alpha)$
	$y' := x \cdot \sin(\alpha) + y \cdot \cos(\alpha)$
Affine Transformation	$x' := k_1 \cdot x + k_2 \cdot y + k_3$
	$y' := k_4 \cdot x + k_5 \cdot y + k_6$

Bei allen Transformationen muß für jedes Zielpixel der Grauwert bestimmt werden anhand der anteiligen Grauwerte in der korrespondierenden Ursprungsfläche.

4.2 Graubildoperationen

Es werden die Grauwerte eines Bildpunktes manipuliert in Abhängigkeit von den Grauwerten der benachbarten Bildpunkte.
Eine solche Operation läßt sich oft durch eine $(2k+1) \times (2k+1)$ Faltungsmatrix M formulieren, deren Koeffizienten mit den Grauwerten g der zu x, y benachbarten Pixel multipliziert werden, die Produkte aufaddiert und mit einer multiplikativen Konstante c gewichtet werden (typischerweise der Kehrwert der Summe der Koeffizienten).

$$\widetilde{g}\ [x,y] := c \cdot \sum_{i=-k}^{k} \sum_{j=-k}^{k} g[x+i, y+j] \cdot M[i,j]$$

Weichzeichner (9 Punkte Gaußfilter)

$$\frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

Zeile:	4	4	5	4	4	5	6	7	8	9	9
gefaltet mit $\frac{1}{4}(1\ 2\ 1)$		4.25	4.5	4.25	4.25	5	6	7	8	8.75	

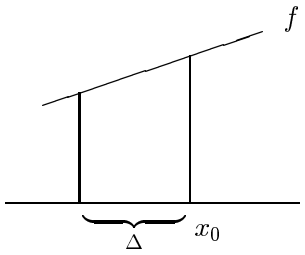
Offenbar können Grauwerte entstehen, die vorher nicht im Bild enthalten waren.
Außerdem verringert sich die Kantensteilheit.

Medianfilter
Beide Effekte vermeidet der nicht-lineare Medianfilter, der den Grauwert an der Stelle x, y durch den Median der umgebenden Grauwerte ersetzt.

Zeile:	4	4	5	4	4	5	6	7	8	9	9
geglättet		4	4	4	4	5	6	7	8	9	

Kantendetektion / Pseudoplastische Darstellung
Betrachte diskrete Ableitung 1. Ordnung:

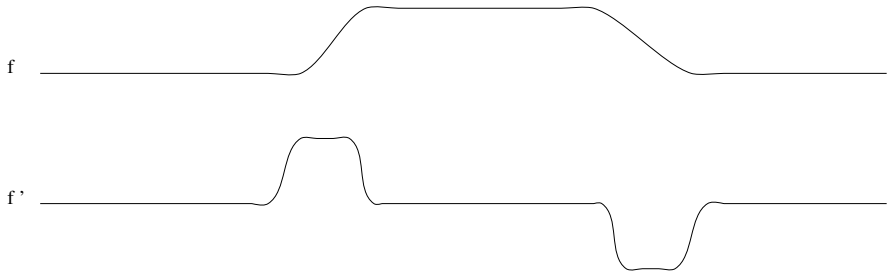
$$f'(x_0) = \lim_{\Delta \to 0} \frac{f(x_0) - f(x_0 - \Delta)}{\Delta}$$



Für Δ	$= 1 \Rightarrow$		
$f'(x_0)$	$=$	$f(x_0) - f(x_0 - 1)$	(Rückwärtsgradient)
$f'(x_0)$	$=$	$f(x_0 + 1) - f(x_0)$	(Vorwärtsgradient)
$f'(x_0)$	$=$	$\frac{f(x_0+1) - f(x_0-1)}{2}$	(Gradient)

$\Rightarrow$ eindimensionale Faltungsmatrix lautet $\frac{1}{2}(-1\ 0\ 1)$

Durch Anwendung dieser Faltung wird eine Helligkeitszunahme durch Weiß, eine Helligkeitsabnahme durch Schwarz signalisiert:



Zeile:	4	4	4	4	5	6	6	6	5	4	4	4	4
gefaltet:		0	0	0.5	1	0.5	0	−0.5	−1	−0.5	0	0	

Im 2-dimensionalen Fall entsteht

$$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$$

Prewitt-Operator zur Verstärkung senkrechter Kanten

$$\begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Prewitt-Operator zur Verstärkung waagerechter Kanten

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Sobel-Operator zur Verstärkung senkrechter Kanten

$$\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

Sobel-Operator zur Verstärkung waagerechter Kanten

Scharfzeichner / Kontrastverstärkung
Betrachte diskrete Ableitung 2. Ordnung.

$$\begin{aligned} f'(x_0) &= f(x_0) - f(x_0 - 1) \text{ (Rückwärtsgradient)} \\ f''(x_0) &= f'(x_0 + 1) - f'(x_0) \\ &= [f(x_0 + 1) - f(x_0)] - [f(x_0) - f(x_0 - 1)] \\ &= f(x_0 + 1) - 2 \cdot f(x_0) + f(x_0 - 1) \end{aligned}$$

$\Rightarrow$ eindimensionale Faltungsmatrix lautet $(1 \; -2 \; \; 1)$

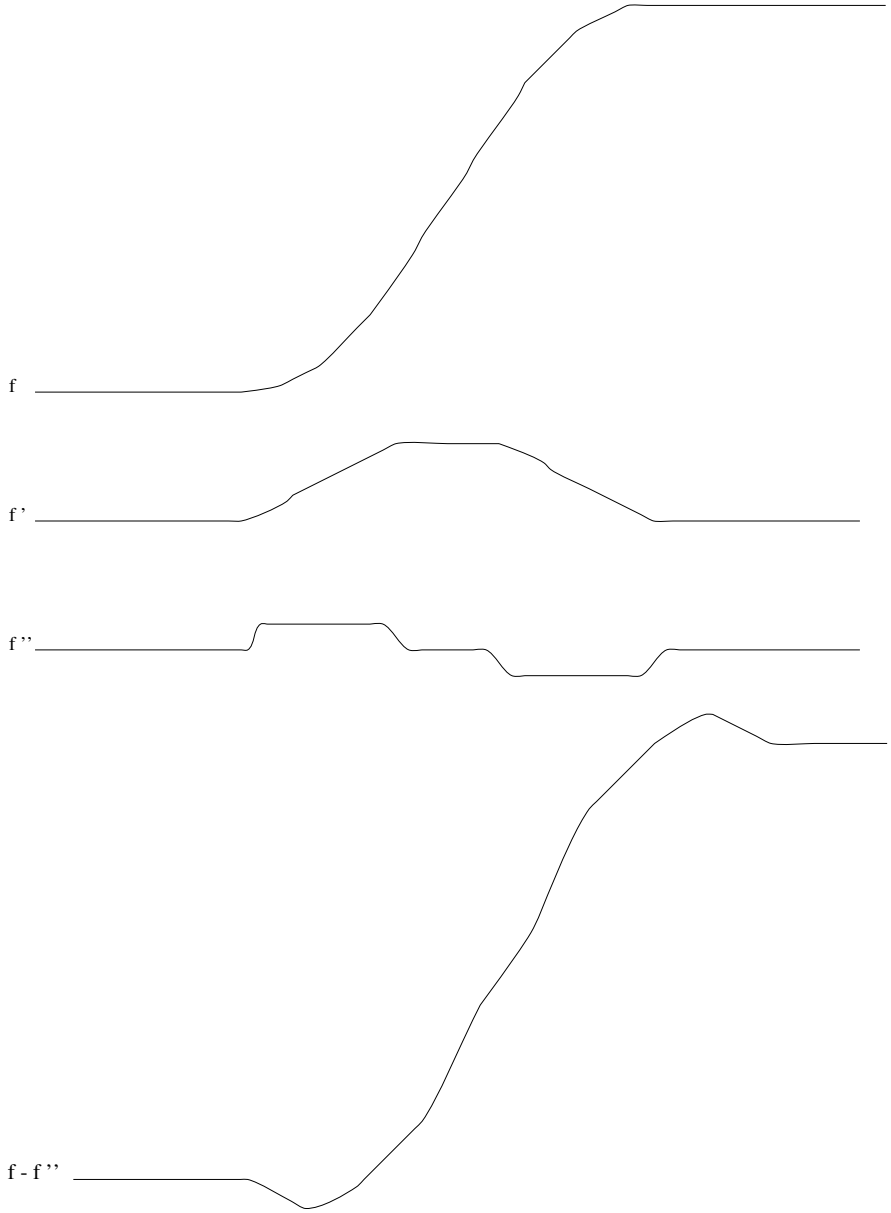
Zur Kontrastverstärkung wird von den ursprünglichen Grauwerten die 2. Ableitung subtrahiert. Dadurch wird bei Beginn eines Helligkeitszuwachses abgedunkelt und bei Ende eines Helligkeitszuwachses aufgehellt.

Im 2-dimensionalen Fall entsteht als Summe der beiden partiellen Ableitungen der Laplace-Operator:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Subtraktion von der (stärker gewichteten) Ausgangsmatrix liefert

$$\frac{1}{4} \begin{pmatrix} -1 & -1 & -1 \\ -1 & +12 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$



Beispiel für eindimensionale Kontrastverstärkung

Gaußscher Weichzeichner

verwackelt

pseudoplastisch

kristallisiert

Gefilterte Versionen eines Grauwertbildes im Format 108 × 157.

4.3 Fourier-Transformation

Fourier (1768-1830) stellte periodische Funktionen als Überlagerung von harmonischen Schwingungen dar.
Sei $f(x)$ eine eindimensionale periodische Funktion mit Periode $T = 2\pi$. Dann läßt sich f darstellen als

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cdot \cos(kx) + b_k \cdot \sin(kx)$$

Die Koeffizienten sind definiert durch

$$a_k := \frac{1}{\pi} \int_{-\pi}^{\pi} f(u) \cdot \cos(ku) du, k = 0, 1, 2, \ldots$$

$$b_k := \frac{1}{\pi} \int_{-\pi}^{\pi} f(u) \cdot \sin(ku) du, k = 0, 1, 2, \ldots$$

und bestimmen die Amplituden der harmonischen Schwingungen.

Motivation:

1. Details im Ortsraum entsprechen den hohen Frequenzen im Frequenzraum, die bei Kompressionsverfahren wegfallen dürfen.
2. Gewisse Bildmanipulationen lassen sich effizienter im Frequenzraum durchführen.

Zum Rechnen nutzt man die Euleridentität aus

$$e^{ikx} = \cos(kx) + i \cdot \sin(kx)$$

Daraus ergibt sich

$$f(x) = \sum_{k=-\infty}^{\infty} c_k \cdot e^{ikx}$$

mit

$$c_k = \frac{1}{\pi} \int f(x) \cdot e^{-ikx} dx$$

Betrachtet man Perioden $-l < x < l,\ l \to \infty$, so erhält man die Fouriertransformierte zu f

$$F(u) := \int f(x) \cdot e^{-iux} dx$$

aus der sich f rekonstruieren läßt durch

$$f(x) := \frac{1}{2\pi} \int F(u) \cdot e^{-ixu} du$$

Die diskrete Fouriertransformation bildet $f(0), f(1), \ldots, f(N-1)$ ab auf

$$F(u) := \frac{1}{N} \sum_{x=0}^{N-1} f(x) \cdot e^{(-i \cdot 2\pi u x)/N}$$

Für den 2-dimensionalen Fall gilt

$$F(u,v) := \frac{1}{M \cdot N} \cdot \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot e^{-i \cdot 2\pi (ux/M + vy/N)}$$

Realteil R und Imaginärteil I lassen sich mit Hilfe der Euler-Identität ausrechnen.

$$R(u,v) := \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot \cos(-2\pi(ux/M + vy/N))$$

$$I(u,v) := \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot \sin(-2\pi(ux/M + vy/N))$$

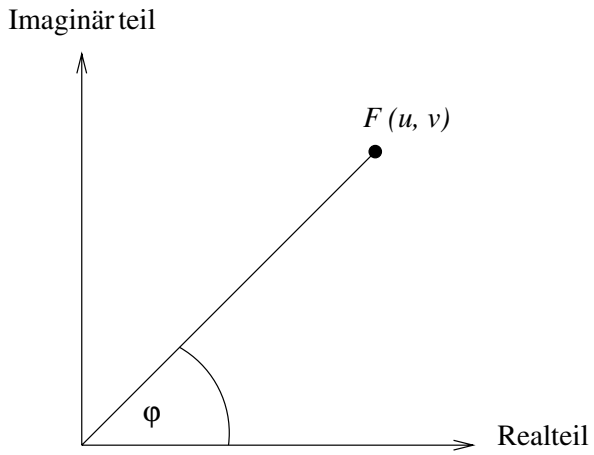
Das Amplitudenspektrum ergibt sich als

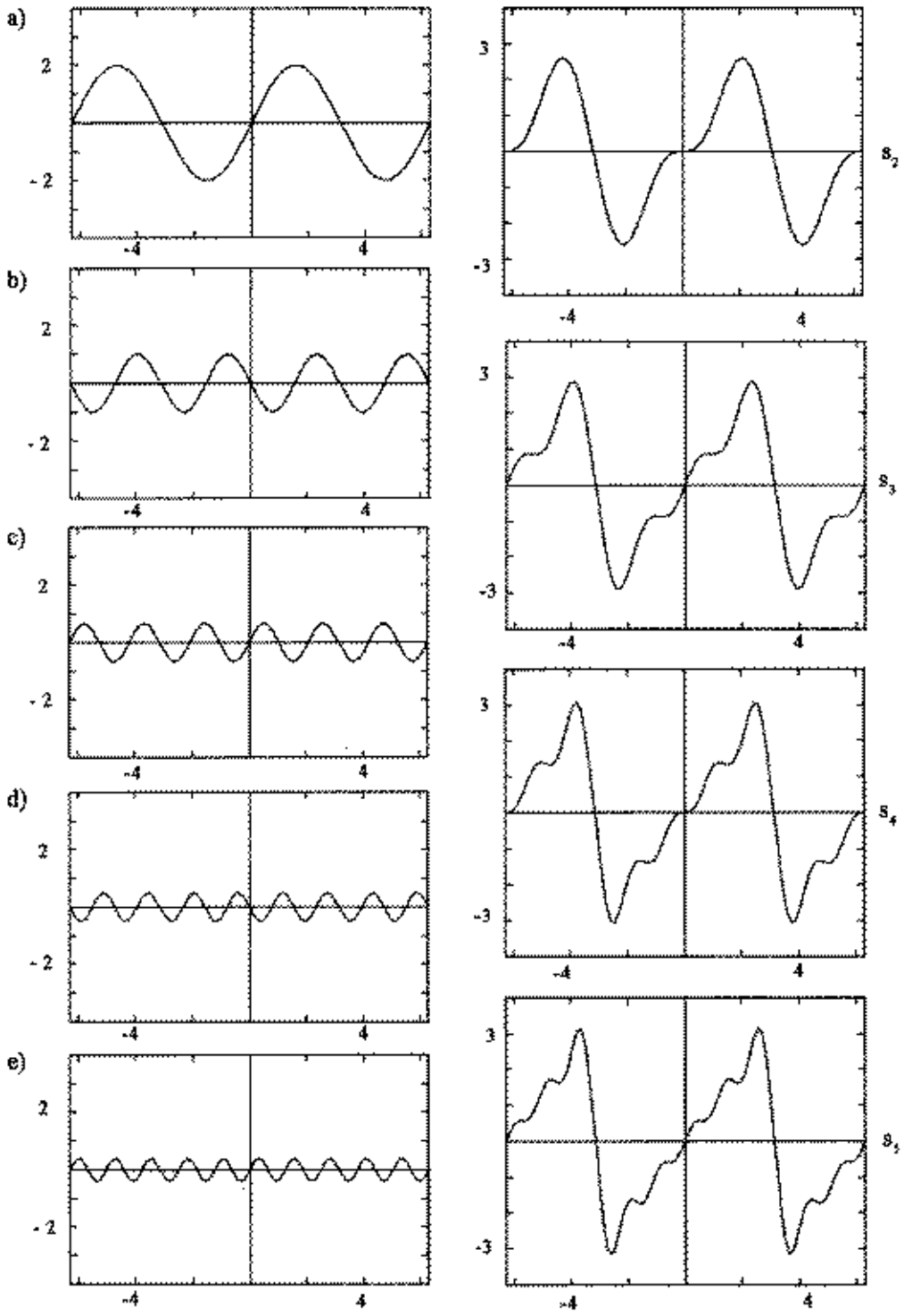
$$|F(u,v)| := \sqrt{R^2(u,v) + I^2(u,v)}$$

und kann durch ein Grauwertbild visualisiert werden (hell = hohe Amplitude, dunkel = niedrige Amplitude, ggf. logarithmische Skalierung, ggf. tiefe Frequenzen in den Bildmittelpunkt verschieben).

Das Phasenspektrum ergibt sich als

$$\varphi(u,v) := arctan(\frac{I(u)}{R(u)})$$



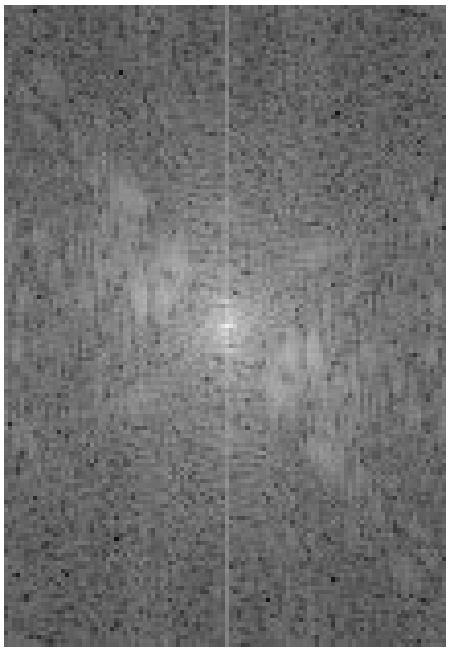


Approximation einer Sägezahnfunktion durch eine Reihe von Sinus-Funktionen

```
for (u=0; u < M; u++)
for (v=0; v < N; v++)
{
    real = 0.0;
    imag = 0.0;

    for (x=0; x < M; x++)
    for (y=0; y < N; y++)
    {
        real += f[x][y] * cos(-2.0 * PI * (u*x/(double)M + v*y/(double)N));
        imag += f[x][y] * sin(-2.0 * PI * (u*x/(double)M + v*y/(double)N));
    };

    amplitude [u][v] = sqrt(real*real + imag*imag);
}
```

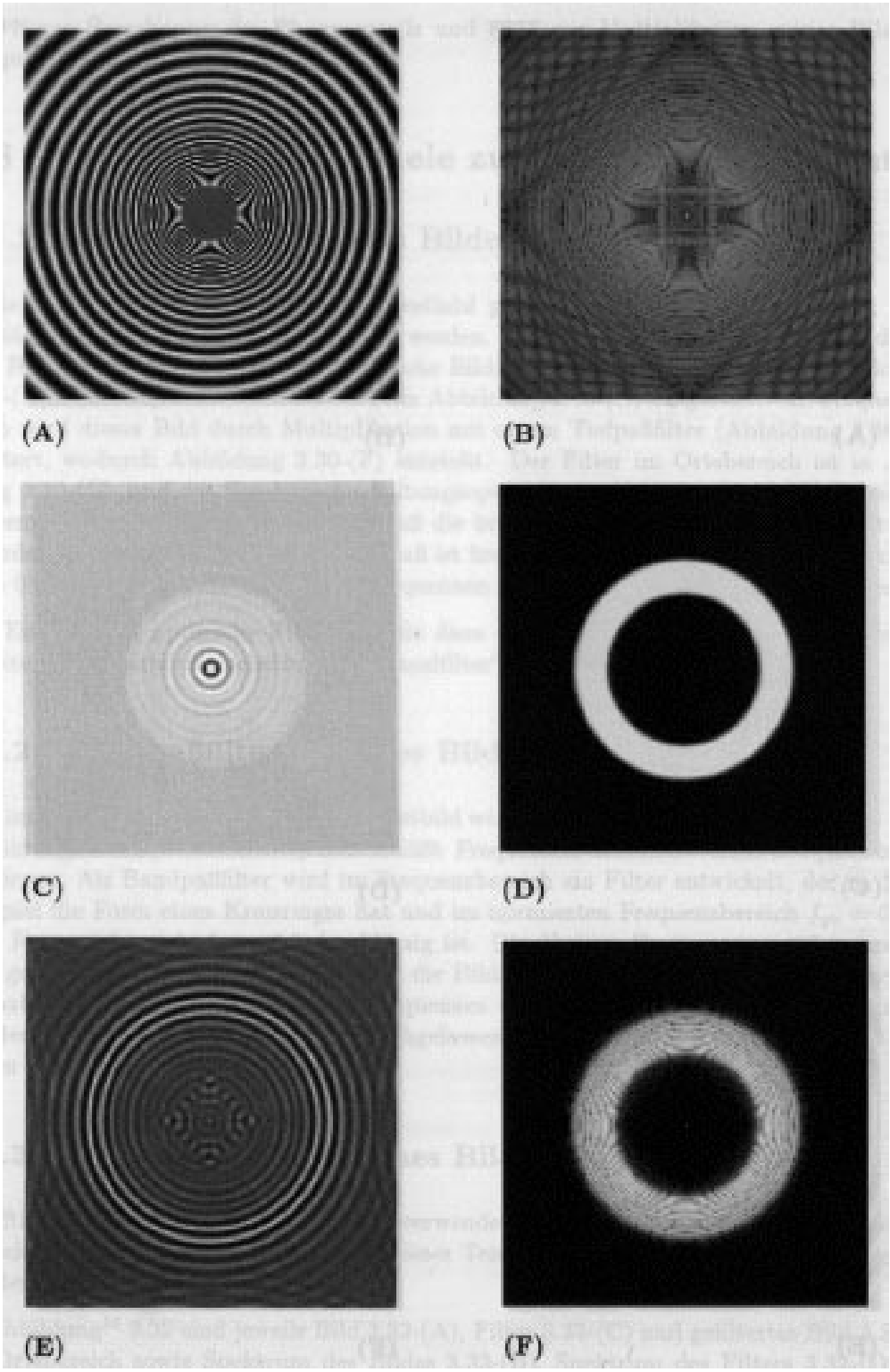
C-Programm zur Berechnung des Fourier-Amplitudenspektrums `amplitude` aus den Grauwerten eines M x N Bildes `f` (mit double-Einträgen)

Ortsbereich

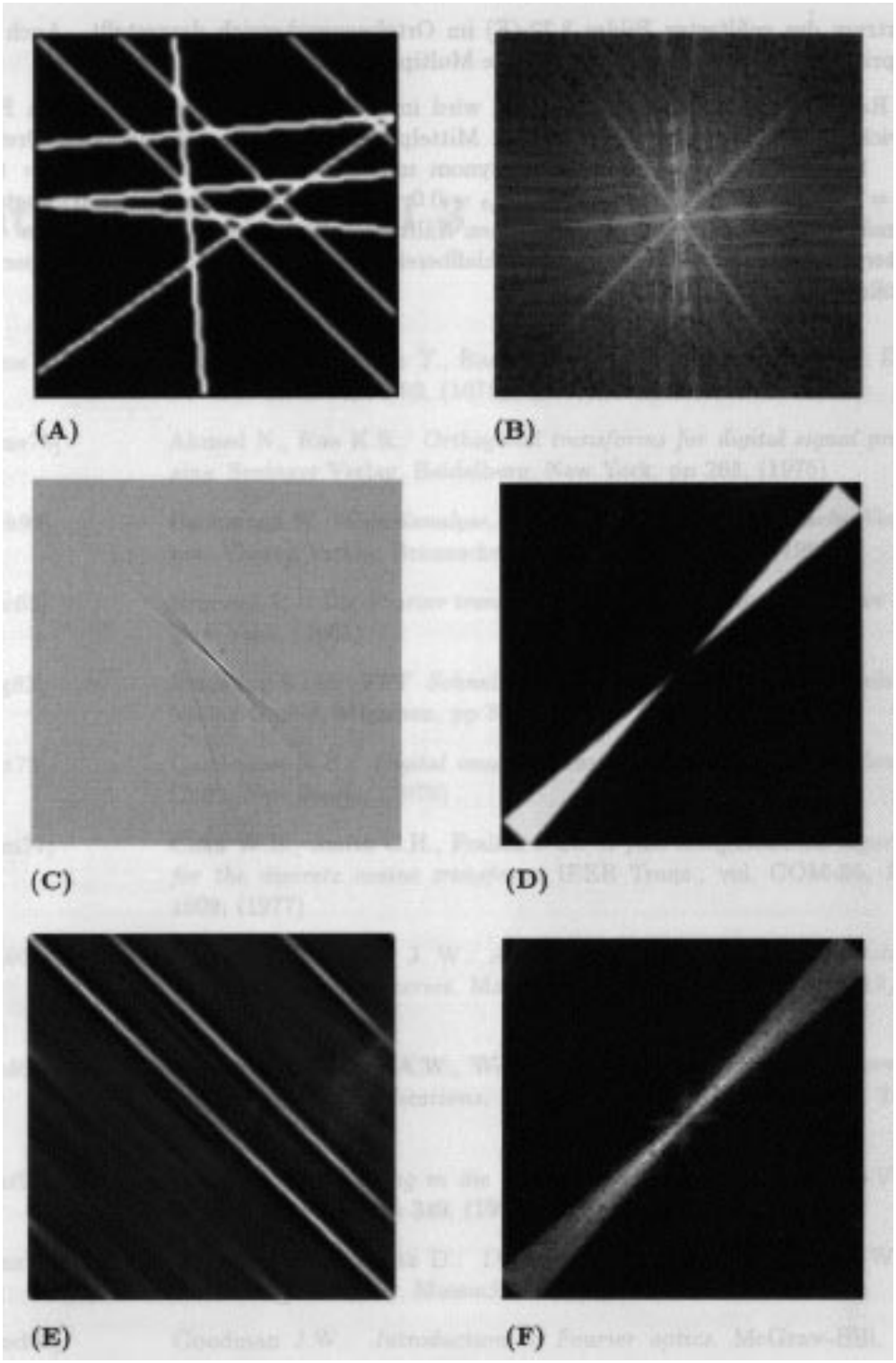


Amplitudenspektrum

Fouriertransformation bei einem Grauwertbild im Format 108 x 157.



Eliminierung der feinen und groben Strukturen im Ortsbereich (A,C,E)
durch Multiplikation mit einem Ring mittlerer Frequenz im Frequenzbereich (B,D,F)

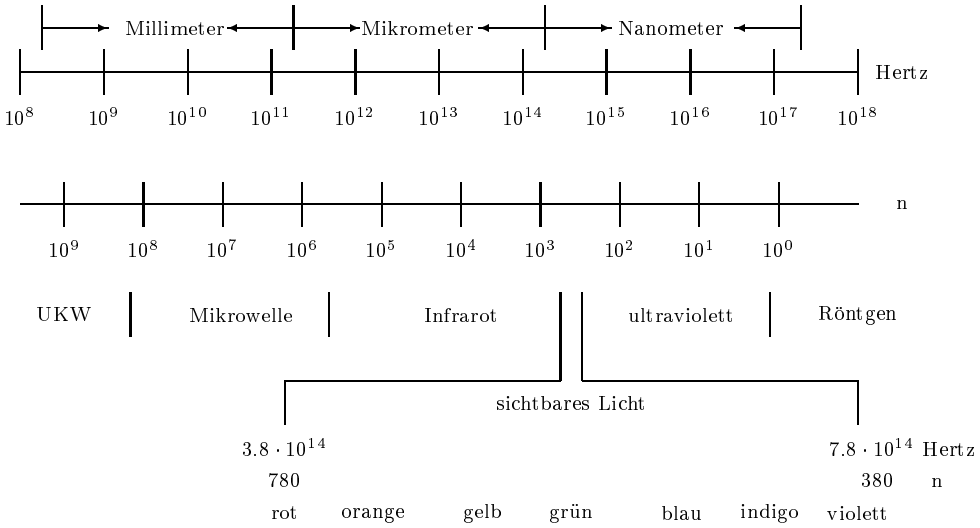


Detektion von Nord/West-Sued/Ost Geraden im Ortsbereich (A,C,E)
durch Multiplikation mit einem geeigneten Keil im Frequenzbereich (B,D,F)

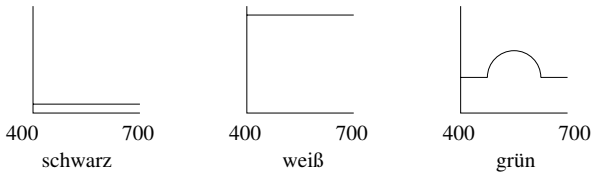
Kapitel 5

Farbbilder

Ein Teil des elektromagnetischen Spektrums wird vom Auge wahrgenommen:



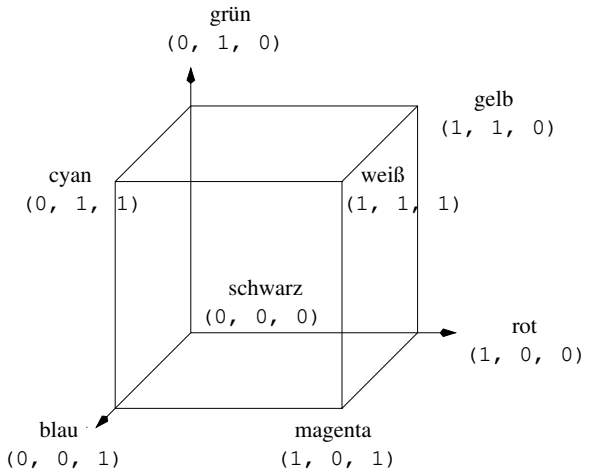
Es gilt: Wellenlänge $\cdot$ Frequenz = Lichtgeschwindigkeit ($= 2.998 \cdot 10^8$ m/sec).
Spektralfarben bestehen aus Licht einer einzigen Wellenlänge.
In der Natur vorkommende Farben bestehen aus Licht, das aus verschiedenen Wellenlängen zusammengesetzt ist. Die Verteilung der Wellenlängen bezeichnet man als Spektrum.

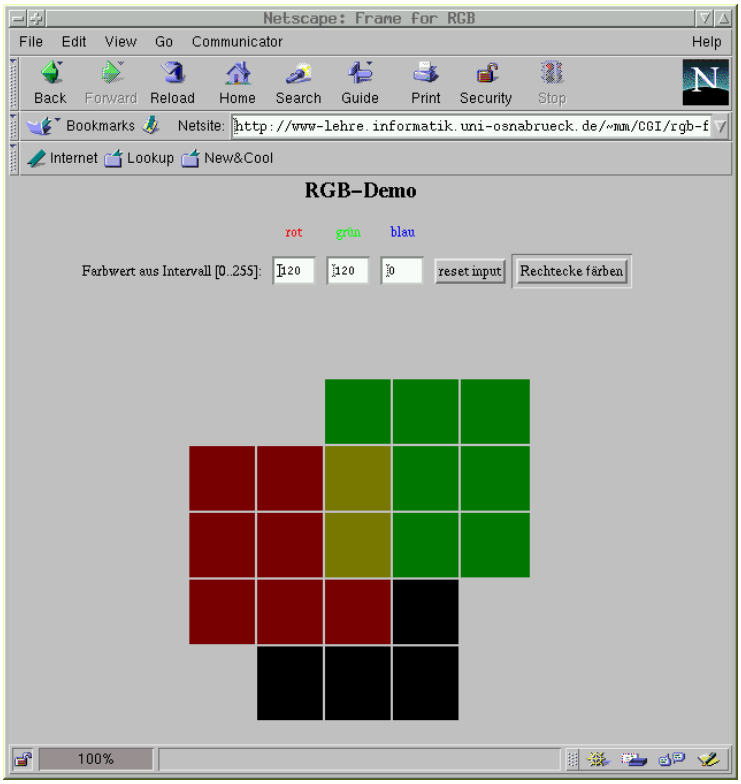


Jedes Pixel eines Farbbildes trägt Informationen zur Beschreibung einer Farbe. Hierbei liegt ein Farbmodell zugrunde, d.h. eine Vereinbarung darüber, wie ein visueller Farbeindruck in digitaler Weise beschrieben werden soll.

5.1 Farbmodelle

RGB-Modell (additiv)
Das RGB-Modell ist entstanden aus der physikalischen Realisierung einer Farbgenerierung am Bildschirm: Ein Elektronenstrahl bringt einen roten, grünen und blauen Phosphorpunkt mit variabler Intensität zum Leuchten. Die Mischung erzeugt beim Betrachter einen bestimmten Farbeindruck. Also wird jeder Farbpunkt durch ein RGB-Tripel beschrieben, welches (typischerweise mit je einem Byte) den Rot-, Grün-, Blauanteil ausdrückt.





RGB-Modell als CGI-Skript

CMY-Modell (subtraktiv)

Bei Farbdrucken empfängt das Auge nur solche Anteile des weißen Lichts, die reflektiert werden. Die zu mischenden Farben entziehen also dem weißen Licht gewisse Bestandteile. Es bietet sich daher ein subtraktives Farbmodell an.

Ein CMY-Tripel beschreibt, wieviel von den Grundfarben Cyan, Magenta, Yellow reflektiert bzw. von den Grundfarben Rot, Grün, Blau absorbiert wird.

Es gilt	0, 0, 0	absorbiert nichts	bleibt weiß
	0, 0, 1	absorbiert blau	bleibt gelb
	0, 1, 0	absorbiert grün	bleibt magenta
	1, 0, 0	absorbiert rot	bleibt cyan
	0, 1, 1	absorbiert cyan	bleibt rot
	1, 0, 1	absorbiert violett	bleibt grün
	1, 1, 0	absorbiert gelb	bleibt blau
	1, 1, 1	absorbiert alles	bleibt schwarz

Beispiel:	(0, 1, 0) Magenta	gemischt mit	(0, 0, 1) Gelb	ergibt	(0, 1, 1) Rot
	(1, 0, 0) Cyan	gemischt mit	(0, 0, 1) Gelb	ergibt	(1, 0, 1) Grün
	(1, 0, 0) Cyan	gemischt mit	(0, 1, 0) Magenta	ergibt	(1, 1, 0) Blau

Es gilt $(R, G, B) = (1, 1, 1) - (C, M, Y)$.

YUV-Modell
Ein Farbwert wird beschrieben durch ein YUV-Tripel, wobei Y die Helligkeit (Luminanz) bezeichnet und U, V Farbdifferenzen (Chrominanz).
Die Helligkeitsempfindung resultiert zu 60 % aus dem Grünanteil, zu 30 % aus dem Rotanteil und zu 10 % aus dem Blauanteil. Genauer

$$Y = 0.2999 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

In den Farbdifferenzen ist die restliche Information codiert:
$U = 0.493 \cdot (B - Y)$
$V = 0.877 \cdot (R - Y)$

Der Vorteil dieses Farbmodells liegt darin begründet, daß in der Y-Komponente das Bild als Matrix von Grauwerten vorliegt und ggf. separat von der Farbinformation weiterverarbeitet werden kann.

5.2 Grafikkarte

Die Leistung einer Grafikkarte wird bestimmt von

- Systembus
- Bildspeicher
- Grafikprozessor
- Digital/Analog-Wandler

Über den Systembus wird die Bilddatei in den Bildspeicher geladen. Von dort wird sie vom Grafikprozessor mit einer gewissen Bildwiederholfrequenz ausgelesen und mit Hilfe des D/A-Wandlers in analoge Spannungswerte zur Ansteuerung des Monitors transformiert.

Ein 800×600 True-Color-Bild erfordert $800 \times 600 \times 3 = 1.440.000$ Bytes Bildspeicher. Um eine ergonomisch günstige Bildwiederholfrequenz von mind. 72 Hz zu erreichen, müssen also $800 \times 600 \times 3 \times 72 \approx 100$ MBytes pro Sekunde transferiert werden.

Ein 1024×768 Hi-Color-Bild erfordert $1024 \times 768 \times 2 = 1.572.864$ Bytes Bildspeicher und bei 72 Hz eine Transferrate von ca. 108 MBytes/sec.

Ein 1280×960 8-Bit-Bild erfordert $1280 \times 960 \times 1 = 1.228.800$ Bytes Bildspeicher und bei 72 Hz eine Transferrate von ca. 84 MBytes/sec.

5.3 Farbtabelle

Ein $n \times m$ True-Color-Bild benötigt $3 \cdot m \cdot n$ Bytes Speicherplatz im Hintergrundspeicher und im Bildwiederholspeicher. Die Zahl der gleichzeitig darstellbaren Farben beträgt $256^3 \approx 16$ Millionen. Dies ist weit mehr, als das menschliche Auge unterscheiden kann. Daher reduziert man den Platzbedarf unter Beibehaltung des visuellen Eindrucks durch eine Farbtabelle. Eine Farbtabelle enthält 2^p Einträge (meistens drei RGB-Bytes) und wird durch Indizes der Länge p Bit referiert. Zur Reduktion des Umfangs einer Bilddatei wird eine maßgeschneiderte Farbtabelle erstellt und zusammen mit den in sie verweisenden Indizes abgespeichert (Palettenbild).

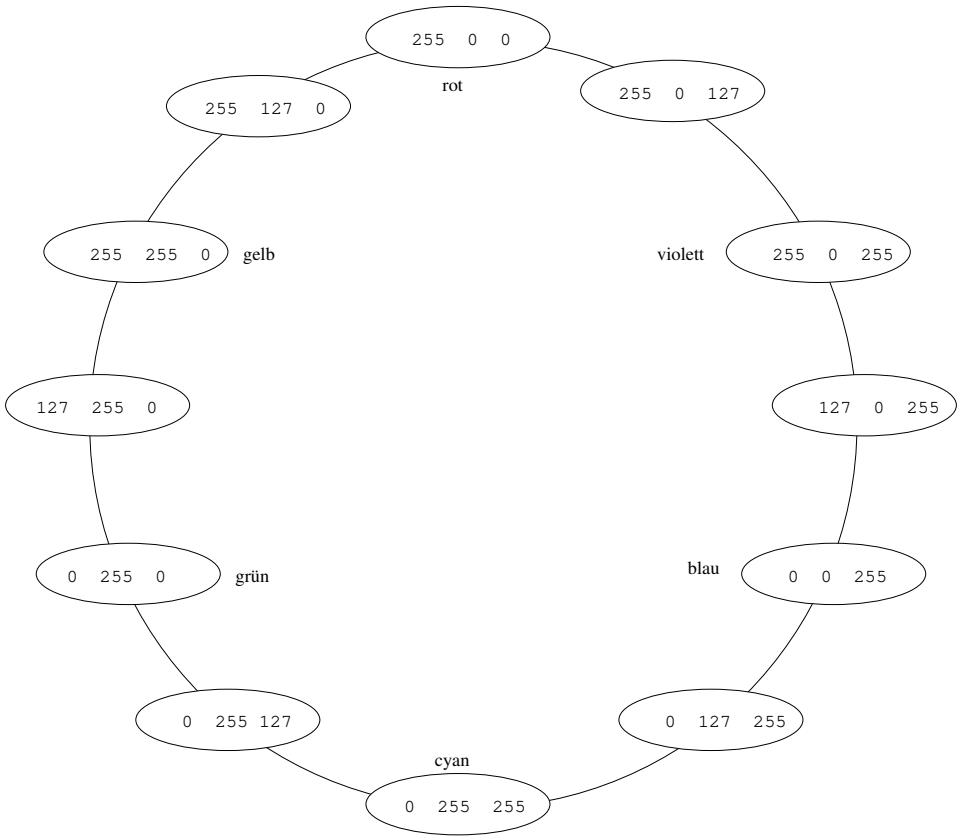
Darstellung eines True-Color-Bildes auf einem 8-Bit-Farbschirm
Es wird eine Farbtabelle initialisiert, die einem $6 \times 6 \times 6$ RGB-Würfel entspricht. D.h. auf insgesamt 216 Einträge verteilt befindet sich das Farbspektrum mit je 6 verschiedenen Rot-, Grün- und Blau-Abstufungen. Jedem RGB-Tripel des True-Color-Bildes wird der Index des nächstgelegenen Farbeintrags zugeordnet. Hierzu wird der Bereich $0 \ldots 255$ in 6 Intervalle partitioniert, denen ein quantisierter Wert zugeordnet ist:

x	0 ... 25	26 ... 76	77... 127	128 ... 178	179 ... 229	230 ... 255
$q(x)$	0	51	102	153	204	255

$$q(x) := \left\lfloor \frac{x + 25}{51} \right\rfloor \cdot 51$$

Hi-Color-Modus
Beim Hi-Color-Modus werden die Rot-, Grün-, Blauanteile eines Pixels durch den Grafikkartentreiber auf $5 + 6 + 5 = 16$ Bit gerundet, d.h., die letzten 3 bzw. 2 Bit werden abgeschnitten. Der resultierende 2 Byte lange Index wird im Bildspeicher abgelegt. Beim Auslesen läßt sich daraus unmittelbar ein Spannungstripel erzeugen, eine Farbtabelle wird nicht benötigt. Es sind also $2^{16} = 65536$ Farben gleichzeitig darstellbar.

Beim Einlesen eines Palettenbildes wird jedem Bildpunkt der 2-Byte Index zugeordnet, der sich nach Rundung des referierten Farbtabelleneintrags ergibt. Danach wird die Farbtabelle nicht mehr benötigt.



12 typische Einträge einer Farbtabelle für einen Bildschirm mit 4 Bit Farbtiefe

5.4 Kompression durch bildbezogene Farbtabelle

Zunächst wird für jede Farbe die Häufigkeit ihres Auftretens ermittelt. Ggf. muß "vorquantisiert" werden von 24 Bit auf 15 Bit (je 5 Bit für Rot, Grün, Blau); hierdurch erhält das Histogramm maximal 32768 Einträge. Sei $d(a,b)$ der Abstand zweier Farbtupel a, b, z.B. $\sqrt{(a_{Rot}-b_{Rot})^2+(a_{Gruen}-b_{Gruen})^2+(a_{Blau}-b_{Blau})^2}$.

Optimaler Algorithmus
Gegeben sei eine Menge F von beobachteten Farben. Gesucht ist eine Menge M von Repräsentanten, so daß der Ausdruck

$$\Delta := \max_{x \in F} \min_{p \in M} d(p,x)$$

minimiert wird, d.h., Δ ist der maximale Abstand, den eine Farbe zu ihrem nächsten Repräsentanten hat.

Da die exakte Bestimmung von M eine exponentielle Laufzeit verursacht, begnügt man sich mit einer Näherungslösung.

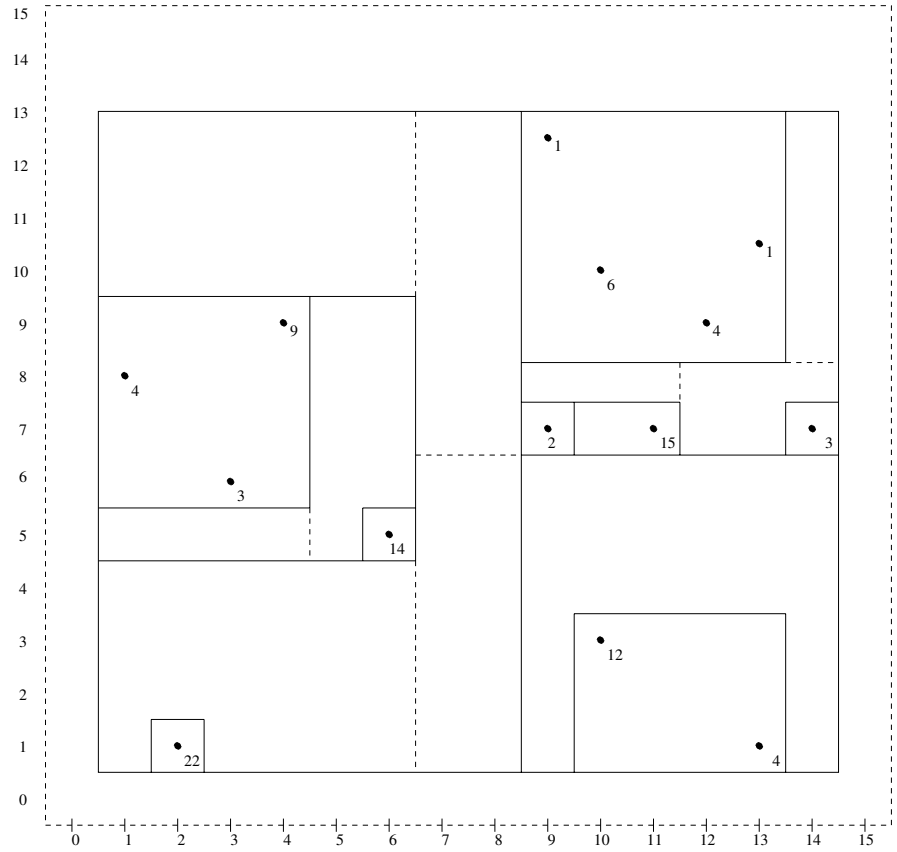
Popularity-Algorithmus (1978)
Wähle die K häufigsten Farben. Nachteil: Selten vorkommende Farben werden schlecht repräsentiert.

Diversity-Algorithmus (*xv*, John Bradley, 1989)
Initialisiere M mit der häufigsten Farbe.

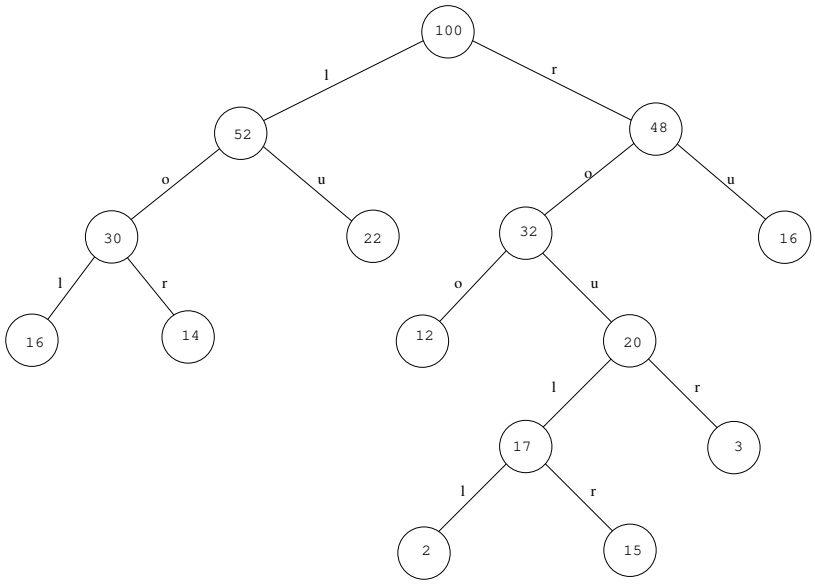
```
for i := 2 to K do
   erweitere M um die Farbe des Histogramms,
   die zu allen Farben aus M den groessten Abstand hat
end
```

Median-Cut (Heckbert, MIT 1980)

```
Initialisiere RGB-Wuerfel mit Haeufigkeiten der beobachteten Farbtupel
Initialisiere Wurzel des Schnittbaums mit Gesamtzahl der Pixel

While noch_nicht_genuegend_Blaetter do

   Waehle Blatt mit der groessten Pixelzahl
   Bestimme umschliessende Box
   Bestimme Achse mit groesstem Wertebereich
   Durchlaufe Box laengs dieser Achse
   Teile am Median in zwei Haelften
   Trage Haelften als Soehne ein
end
Fuer jedes Blatt waehle den Mittelwert aller in ihm liegenden Farben.
```



Beispiel für die Anwendung des Median-Cut-Algorithmus. Zur einfachen Darstellung ist der Farbbaum zweidimensional gewählt. Gezeigt wird die Verteilung der Farbtupel aus dem Bereich $[0 \ldots 15] \times [0 \ldots 15]$ in einem Bild mit $10 \times 10 = 100$ Pixeln. Eingetragen sind an der Position x/y die Häufigkeit des Farbtupels x, y. Schnittlinien sind gestrichelt, umschließende Boxen durchgezogen gezeichnet.



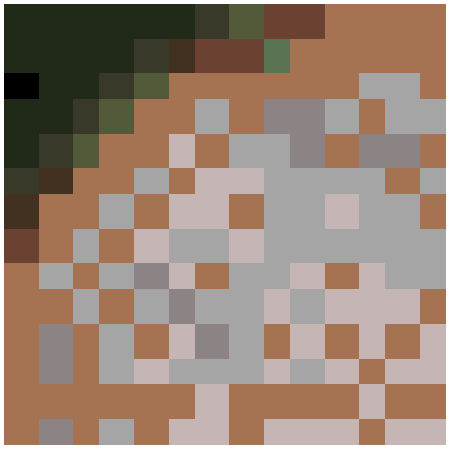
Schnittbaum nach Anwendung des Median-Cut-Algorithmus. Die Knoten sind markiert mit der Anzahl der noch zu quantisierenden Pixel, an den Kanten ist vermerkt, ob es sich um einen Links/Rechts- oder um einen Oben/Unten-Schnitt handelt.

Floyd-Steinberg-Dithering (1975)
Der bei Verwendung einer Farbtabelle verursachte Fehler beim Quantisieren eines Pixels wird auf die Nachbarpixel verteilt (bevor diese quantisiert werden).

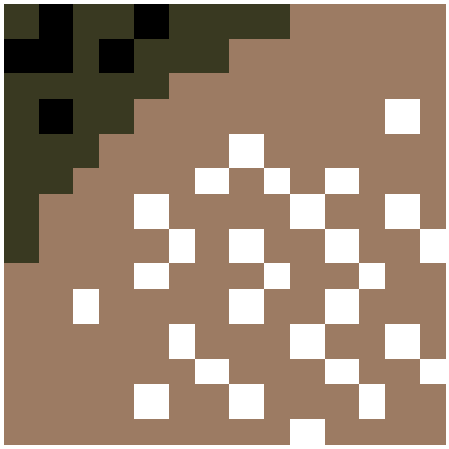
```
for (i = 0; i < M; i++)
for (j = 0; j < N; j++)
{
   x = f[i][j];   /* hole Original-Farbe */
   k = p(x);      /* finde naechsten Repraesentant */
   q[i][j] = k;   /* zeichne quantisiertes Pixel */
   e = d(x, k);   /* bestimme Fehlerabstand */
   f [i][j + 1]     = f[i][j + 1]     + e * 3.0/8.0;
   f [i + 1][j]     = f[i + 1][j]     + e * 3.0/8.0;
   f [i + 1][j + 1] = f[i + 1][j + 1] + e * 1.0/4.0;
}
```

24 Bit pro Pixel,
16 Mill. Farben

8 Bit pro Pixel,
256 Farben



4 Bit pro Pixel,
16 Farben



2 Bit pro Pixel,
4 Farben

Original und quantisierte Versionen einer 14×14 Ausschnittvergrößerung, erstellt von einem 814×517 True-Color-Bild. Die Zahl der Farben bezieht sich jeweils auf das Gesamtbild. Verwendet wurde der Median-Cut-Algorithmus mit anschließendem Floyd-Steinberg-Dithering.

5.5 Kompression nach JPEG

Die **J**oint **P**hotographic **E**xpert **G**roup bildete sich aus Mitgliedern der Standardisierungsgremien CCITT (Consultative Committee for International Telephone and Telegraph) und ISO (International Standardization Organization). JPEG wurde schließlich zum Namen für den Standard selbst.

Zunächst wird das RGB-Bild in den YUV-Raum transformiert. Da das Auge für Helligkeitssprünge sensitiver ist als für Farbdifferenzen, kann man nun die Y-Matrix in der vollen Auflösung belassen und in den U, V-Matrizen jeweils 4 Pixel mitteln (4:1:1 Subsampling).

Für je 4 Originalpixel mit insgesamt 12 Bytes werden nun $4 + 1 + 1 = 6$ Bytes benötigt (pro Bildpunkt also $6 \cdot 8/4 = 12$ Bit). Die Reduktion beträgt 50 %.

Werden je 16 Originalpixel bzgl. der Farbwerte gemittelt, so werden für 16 Originalpixel statt 48 Bytes nur noch $16 + 1 + 1 = 18$ Bytes benötigt (pro Bildpunkt also $18 \cdot 8/16 = 9$ Bit). Die Bilddatei schrumpft auf 37.5 %.

Beim JPEG-Verfahren werden nun die drei Matrizen in Blöcke mit 8×8 Abtastwerten aufgeteilt. Anschließend durchlaufen die Blöcke folgende Schritte:

1. Diskrete Cosinus Transformation
2. Rundung der Frequenzkoeffizienten
3. Lauflängenkodierung der quantisierten Werte
4. Huffmancodierung der Lauflängenbeschreibung.

Um aus dem komprimierten Bild das Original zu rekonstruieren, werden die Schritte in umgekehrter Reihenfolge und inverser Funktionalität durchlaufen.

Durch die Wahl der Rundungstabelle läßt sich der Tradeoff zwischen Qualität und Kompression beliebig steuern. Ein typisches Farbbild läßt sich ohne für das Auge sichtbare Artefakte auf 10 % seiner Originalgröße reduzieren. Eine Reduktion auf 5 % verursacht oft nur leichte, kaum wahrnehmbare Verzerrungen.

DCT (Diskrete Cosinus Transformation)
Die diskrete Cosinus-Transformation ist ein Spezialfall der Fouriertransformation. Es wird ausgenutzt, daß für "gerade" Funktionen (d.h. $f(-x) = f(x)$) der Sinus-Term mit seinem imaginären Anteil wegfällt. Ein zweidimensionaler Bildbereich läßt sich durch Spiegelung an der y-Achse künstlich gerade machen.

$$s[u,v] := \frac{1}{4} \cdot c_u \cdot c_v \cdot \sum_{x=0}^{7} \sum_{y=0}^{7} f[x,y] \cdot \cos\frac{(2x+1)\cdot u \cdot \pi}{16} \cdot \cos\frac{(2y+1)\cdot v \cdot \pi}{16}$$

$$c_u, c_v := \begin{cases} \frac{1}{\sqrt{2}} & \text{für } u,v = 0 \\ 1 & \text{sonst} \end{cases}$$

Hierdurch wird eine 8×8 Ortsmatrix in eine 8×8 Frequenzmatrix transformiert.

Mit Hilfe der inversen DCT lassen sich die Originalwerte rekonstruieren.

$$f[x,y] := \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} c_u \cdot c_v \cdot s[u,v] \cdot \cos\frac{(2x+1)\cdot u \cdot \pi}{16} \cdot \cos\frac{(2y+1)\cdot v \cdot \pi}{16}$$

Für die Bildverarbeitung wird der Pixelwertebereich 0..255 in das symmetrische Intervall $-128..127$ verschoben. Der Wertebereich von s liegt dann im Intervall $-1024..+1023$. Der errechnete Koeffizient s_{00} entspricht dem Anteil der Frequenz null in beiden Achsen und wird DC-Koeffizient (Gleichspannungsanteil) bezeichnet. Die übrigen Koeffizienten werden AC-Koeffizienten (Wechselspannungsanteil) genannt. Z.B. bezeichnet s_{77} die höchste, in beiden Richtungen auftretende Frequenz.

Die Eingangsmatrix M läßt sich auch durch $T \cdot M \cdot T'$ transformieren mit Hilfe der Matrix T:

```
0.353553    0.353553    0.353553    0.353553    0.353553    0.353553    0.353553    0.353553
0.490393    0.415735    0.277785    0.097545   -0.097545   -0.277785   -0.415735   -0.490393
0.461940    0.191342   -0.191342   -0.461940   -0.461940   -0.191342    0.191342    0.461940
0.415735   -0.097545   -0.490393   -0.277785    0.277785    0.490393    0.097545   -0.415735
0.353553   -0.353553   -0.353553    0.353553    0.353553   -0.353553   -0.353553    0.353553
0.277785   -0.490393    0.097545    0.415735   -0.415735   -0.097545    0.490393   -0.277785
0.191342   -0.461940    0.461940   -0.191342   -0.191342    0.461940   -0.461940    0.191342
0.097545   -0.277785    0.415735   -0.490393    0.490393   -0.415735    0.277785   -0.097545
```

Quantisierung
Die errechnete Matrix hat von links oben nach rechts unten Werte abnehmender Größe. Da die Werte rechts unten den hohen, eher unwichtigen Frequenzen entsprechen, werden alle Einträge mit Faktoren zunehmender Größe dividiert.

$$r[u,v] := \frac{s[u,v]}{q[u,v]}$$

95	88	87	95	88	95	95	95
143	144	151	151	153	170	183	181
153	151	162	166	162	151	126	117
143	144	133	130	143	153	159	175
123	112	116	130	143	147	162	189
133	151	162	166	170	188	166	128
160	168	166	159	135	101	93	98
154	155	153	144	126	106	118	133

Bildmatrix

↓

93	2	-8	-7	3	1	1	-2
-38	-58	11	17	-3	5	5	-3
-84	63	-1	-17	2	7	-4	-0
-51	-37	-10	13	-10	5	-1	-4
-85	-42	50	-8	18	-5	-1	1
-63	66	-13	-1	2	-6	-2	-2
-16	14	-37	18	-12	4	3	-3
-53	31	-7	-10	23	-0	2	2

DCT-Koeffizienten

→

31	0	-1	0	0	0	0	0
-7	-8	1	1	0	0	0	0
-12	7	0	-1	0	0	0	0
-5	-3	0	0	0	0	0	0
-7	-3	3	0	0	0	0	0
-4	4	0	0	0	0	0	0
-1	0	-1	0	0	0	0	0
-3	1	0	0	0	0	0	0

quantisierte DCT-Koeffizienten

↑

98	95	91	89	90	95	101	106
140	143	149	156	163	167	168	167
146	149	154	159	159	151	137	126
149	142	136	137	145	156	163	166
119	117	118	125	140	157	170	176
137	147	160	170	172	166	157	150
166	167	164	152	132	112	99	93
151	153	150	139	125	118	119	123

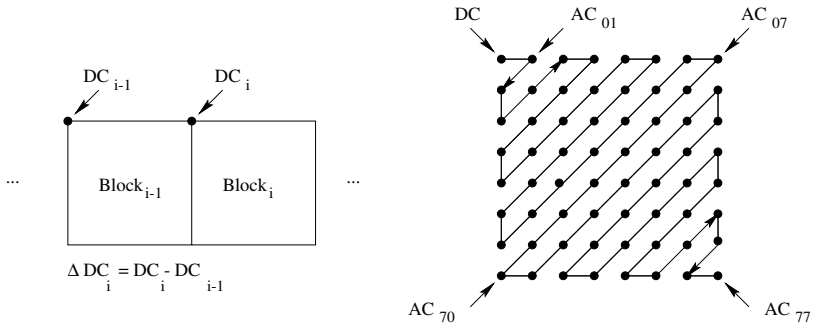
rekonstruierte Bildmatrix

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

Quantisierungsmatrix

Entropiekodierung
Die *DC*-Koeffizienten benachbarter Blöcke unterscheiden sich nur wenig und werden daher als Differenz zum Vorgängerblock übertragen.

Die *AC*-Koeffizienten werden zunächst in eine Zick-Zack-Sequenz umgeordnet:



Die AC-Koeffizienten längs dieses Weges bis zum letzten Eintrag ungleich Null werden als Folge von Paaren beschrieben:

Symbol 1: Länge der ununterbrochenen Folge von Nullen vor diesem Wert (Runlength)
Anzahl der Bits, die zur Darstellung des Wertes erforderlich sind.
Symbol 2: der Wert selbst

Der quantisierte DCT-Koeffizient AC_{70} aus vorigem Beispiel wird beschrieben als Tupel $< (11, 2), -3 >$, denn vor ihm in der Zickzacksequenz stehen 11 Nullen, sein Wert kann mit 2 Bit codiert werden, und sein Wert beträgt -3.

Für das Symbol 1 gibt es Häufigkeitsverteilungen, aus denen sich ein Huffman-Code konstruieren läßt. Für das Symbol 2 wählt man ein 2-er Komplement, welches berücksichtigt, daß bei angekündigten N Bits nur solche Zahlen kodiert werden müssen, für die $N - 1$ Bits nicht ausreichen.

Länge/Bitzahl	Codierung
0/0 (EOB)	1010
0/1	00
0/2	01
0/3	100
0/4	1011
0/5	11010
0/6	1111000
0/7	11111000
0/8	1111110110
0/9	1111111110000010
0/10	1111111110000011
1/1	1100
1/2	11011
1/3	1111001
⋮	⋮
2/1	11100
2/2	11111001
2/3	1111110111
⋮	⋮
3/1	111010
3/2	111110111
3/3	111111110101
⋮	⋮
11/1	1111111001
11/2	1111111111010000
⋮	⋮
15/10	1111111111111110

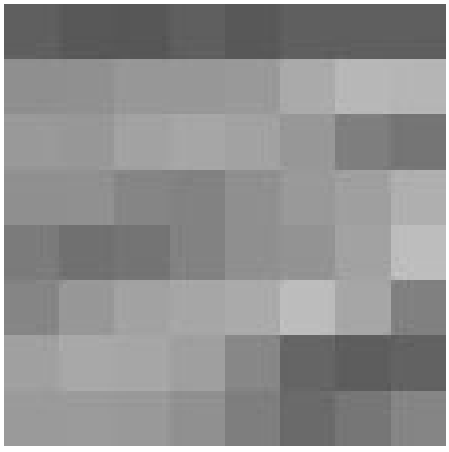
Huffman Codierung
für Symbol 1

Wert	Codierung
15	1111
14	1110
13	1101
12	1100
11	1011
10	1010
9	1001
8	1000
7	111
6	110
5	101
4	100
3	11
2	01
1	1
-1	0
-2	10
-3	00
-4	011
-5	010
-6	001
-7	000
-8	0111
-9	0110
-10	0101
-11	0100
-12	0011
-13	0010
-14	0001
-15	0000

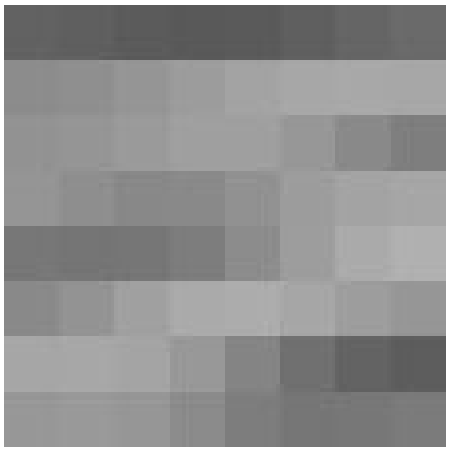
Komplement-Codierung
für Symbol 2

Symbol 1	Symbol 2	Huffman für Symbol 1	2-er Komplement für Symbol 2
(1, 3)	-7	1111001	000
(0, 4)	-12	1011	0011
(0, 4)	-8	1011	0111
(0, 1)	-1	00	0
(1, 1)	1	1100	1
(0, 3)	7	100	111
(0, 3)	-5	100	010
(0, 3)	-7	100	000
(0, 2)	-3	01	00
(1, 1)	1	1100	1
(3, 1)	-1	111010	0
(1, 2)	-3	11011	00
(0, 3)	-4	100	011
(0, 1)	-1	00	0
(0, 3)	4	100	100
(0, 2)	3	01	11
(11, 2)	-3	1111111111010000	00
(0, 1)	1	00	1
(0, 1)	-1	00	0
EOB		1010	

Symbole und ihre Kodierung für die im Beispiel vorgestellten quantisierten AC-Koeffizienten



Ausgangsbildmatrix



rekonstruierte Bildmatrix

8 x 8 Matrix vor und nach der JPG-Kompression.

motel.tif
100%

motel80.jpg
9%

motel40.jpg
5%

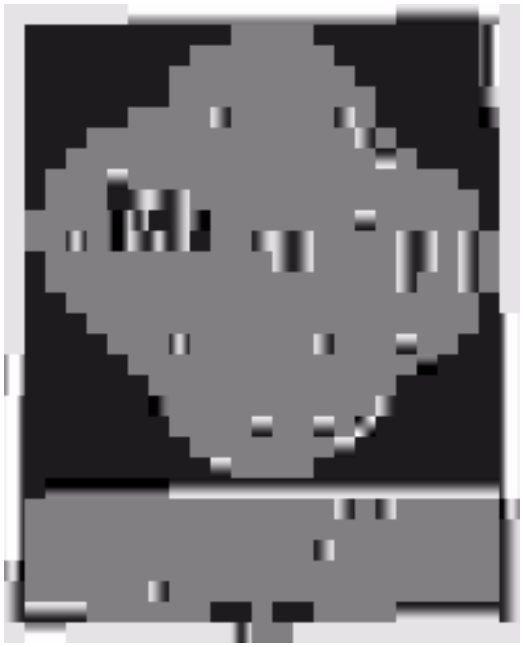
motel20.jpg
4%

Kompression nach dem JPEG-Verfahren. Platzbedarf in % bezogen auf tif-Datei.

motel10.jpg
3%

motel05.jpg
2%

motel02.jpg
1.5%



motel01.jpg
1%

Kompression nach dem JPEG-Verfahren. Platzbedarf in % bezogen auf tif-Datei.

95	88	87	95	88	95	95	95
143	144	151	151	153	170	183	181
153	151	162	166	162	151	126	117
143	144	133	130	143	153	159	175
123	112	116	130	143	147	162	189
133	151	162	166	170	188	166	128
160	168	166	159	135	101	93	98
154	155	153	144	126	106	118	133

Dezimaldarstellung der Bildbereich-Matrix

01011111	01011000	01010111	01011111	01011000	01011111	01011111	01011111
10001111	10010000	10010111	10010111	10011001	10101010	10110111	10110101
10011001	10010111	10100010	10100110	10100010	10010111	01111110	01110101
10001111	10010000	10000101	10000010	10001111	10011001	10011111	10101111
01111011	01110000	01110100	10000010	10001111	10010011	10100010	10111101
10000101	10010111	10100010	10100110	10101010	10111100	10100110	10000000
10100000	10101000	10100110	10011111	10000111	01100101	01011101	01100010
10011010	10011011	10011001	10010000	01111110	01101010	01110110	10000101

8-Bit-Codierung für Bildbereich-Matrix

00011111

quantisierter DC-Koeffizient

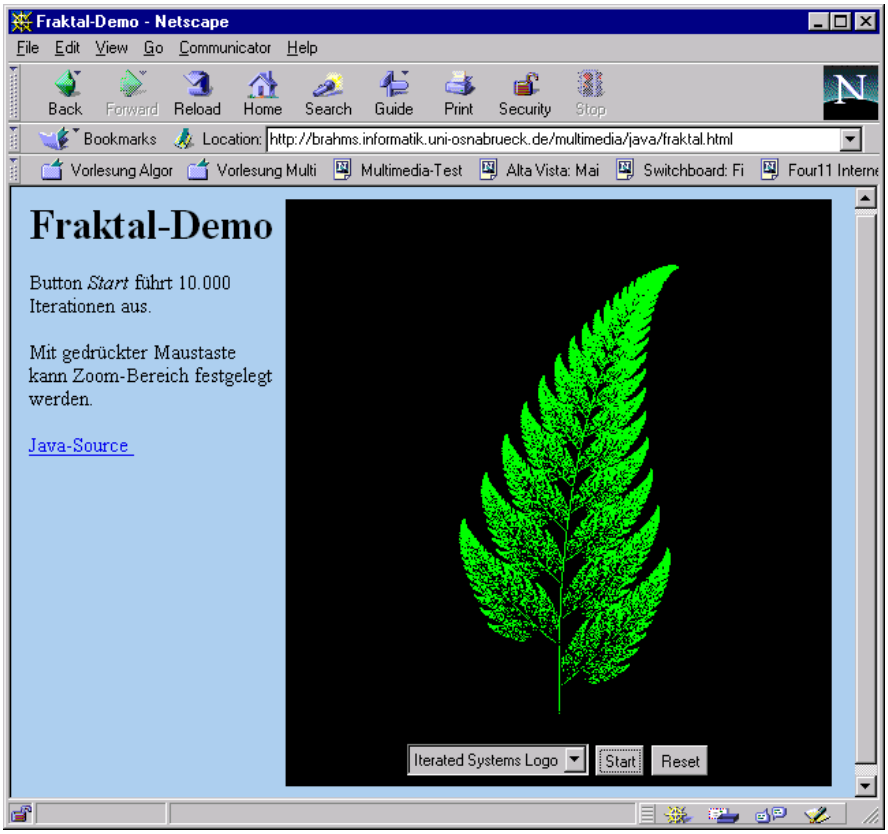
111100100010110011101101110001100110011110001010000001001100111101001101100100011000100100011111111111111010000000010001010

Huffman-Codierung für quantisierte AC-Koeffizienten

5.6 Fraktale Kompression

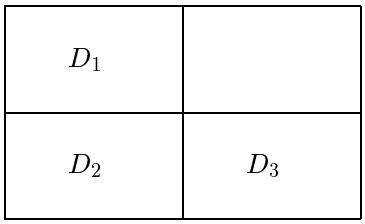
Iterierte Funktionssysteme sind in der Lage, mit wenigen Regeln komplexe, natürlich aussehende Bilder zu erzeugen. Hierbei wird eine Folge von Punkten im $\mathbf{R}^2$ durch fortgesetzte Anwendung von affinen Transformationen durchlaufen. Jede Transformation ist definiert durch eine 2×2- Deformationsmatrix A, einen Translationsvektor b und eine Anwendungswahrscheinlichkeit p, wodurch ein Punkt $\overline{x}$ abgebildet wird auf $A\overline{x} + b$. Zum Beispiel erzeugen folgende 4 Regeln das Blatt eines Farns:

	A	b	p
w_1	$\begin{pmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 10.6 \end{pmatrix}$	85 %
w_2	$\begin{pmatrix} 0.20 & -0.26 \\ 0.23 & 0.22 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 10.6 \end{pmatrix}$	7 %
w_3	$\begin{pmatrix} -0.15 & 0.28 \\ 0.26 & 0.24 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 4.4 \end{pmatrix}$	7 %
w_4	$\begin{pmatrix} 0.00 & 0.00 \\ 0.00 & 0.16 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$	1 %

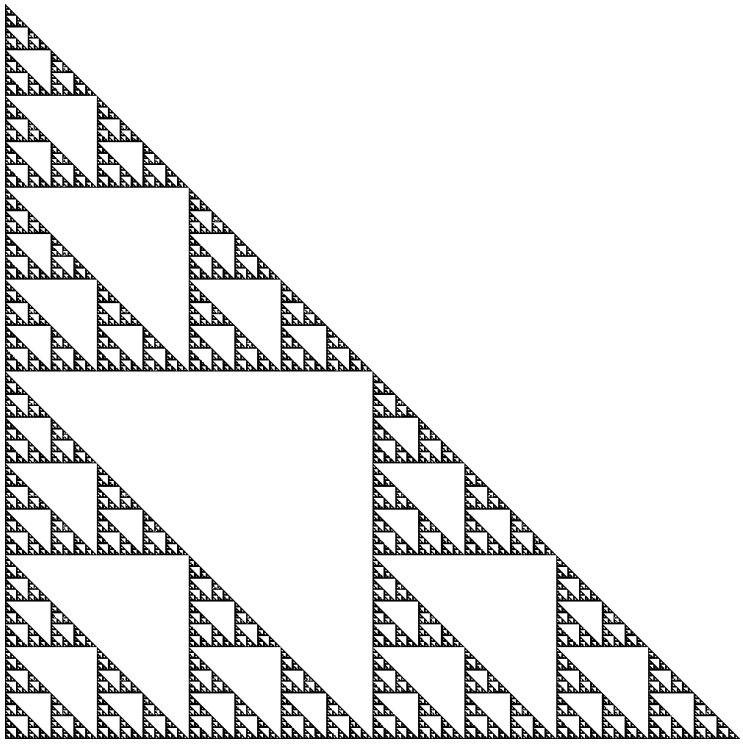


Blatt eines Farns

Eine alternative Sichtweise zur Definition eines fraktalen Bildes verlangt die Selbstähnlichkeit von Teilen des Bildes mit dem Ganzen.



Beispielsweise sei ein Rechteck R gesucht, so daß sich sein Inhalt, jeweils auf ein Viertel verkleinert, im linken oberen, linken unteren und rechten unteren Quadranten wiederfindet. Beginnend mit einem beliebigen Rechteckinhalt werden die 3 Quadranten so lange als skalierte Versionen des Gesamtrechtecks ersetzt, bis wir nahe am Fixpunkt dieser Iterationen angelangt sind. Das Ergebnis ist das sogenannte *Sierpinsky-Dreieck*.



Sierpinsky-Dreieck

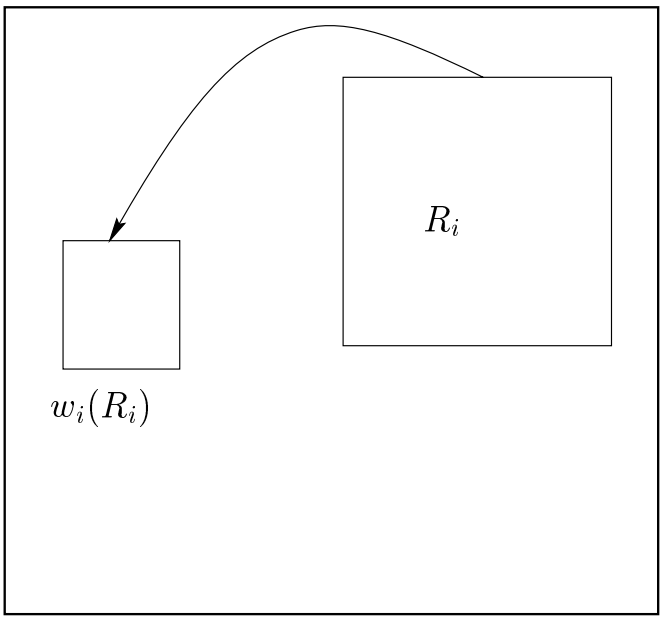
Im Gegensatz zu einem Pixelbild mit fester Auflösung in horizontaler und vertikaler Richtung läßt sich in ein fraktal erzeugtes Bild beliebig reinzoomen, da sein Inhalt durch eine auflösungsunabhängige Rechenvorschrift definiert ist.
Ziel der fraktalen Kompression ist es, in einer gegebenen Bildvorlage Selbstähnlichkeiten zu erkennen und diese durch ein System von affinen Transformationen zu beschreiben. Beim Entpacken können dann die Auswirkungen dieser Transformationen in beliebiger Detaillierung berechnet werden.

Zur fraktalen Kompression eines S/W-Bildes G werden zunächst die schwarzen Bildteile durch rechteckige, disjunkte Domain-Blöcke $D_1, D_2, \ldots, D_n$ überdeckt. Dann wird zu jedem Domain-Block D_i ein zu ihm selbstähnlicher Range-Block R_i gesucht, d.h. ein Teilbereich R_i des Bildes G und eine affine, kontraktive Abbildung w_i mit

$$w_i : R_i \to D_i$$

$$w_i(R_i \cap G) \approx D_i \cap G$$

d.h., das durch w_i erzeugte Abbild des Inhaltes von R_i liegt sehr nah (im Sinne einer geeigneten Metrik) am Inhalt von D_i.

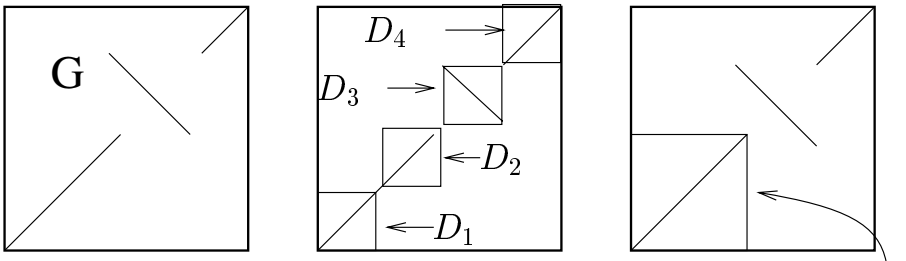


Domain-Block D_i wird angenähert durch $w_i(R_i)$

In einer praktischen Implementierung sind alle D_i Teile einer rechteckigen Kachelung von G. Wegen der Kontraktivität der affinen Abbildungen sind die Regionen R_i (ggf. überlappende) Rechtecke mit jeweils doppelter Kantenlänge. Bei der Wahl der w_i beschränkt man sich auf jeweils eine der 8 elementaren Verformungen mithilfe von Spiegelungen an den x-, y- und Diagonal-Achsen sowie Rotationen um 0, 90, 180 und 270 Grad.

Symmetrie	Matrix	Beschreibung
0	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	Identität
1	$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$	Spiegelung an y-Achse
2	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	Spiegelung an x-Achse
3	$\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$	180° Drehung
4	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	Spiegelung an $(y = x)$-Achse
5	$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$	90° Drehung
6	$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$	270° Drehung
7	$\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$	Spiegelung an $(y = -x)$-Achse

Die 8 möglichen Transformationsmatrizen



Ausgangsbild Domain-Bl"ocke $R_1 = R_2 = R_3 = R_4$

Map#	D_x	D_y	R_x	R_y	Symmetrie
1	0	0	0	0	0
2	4	4	0	0	0
3	8	8	0	0	2
4	12	12	0	0	0

Ausgangsbild, Domain-Blöcke, Range-Blöcke, 4 affine Abbildungen

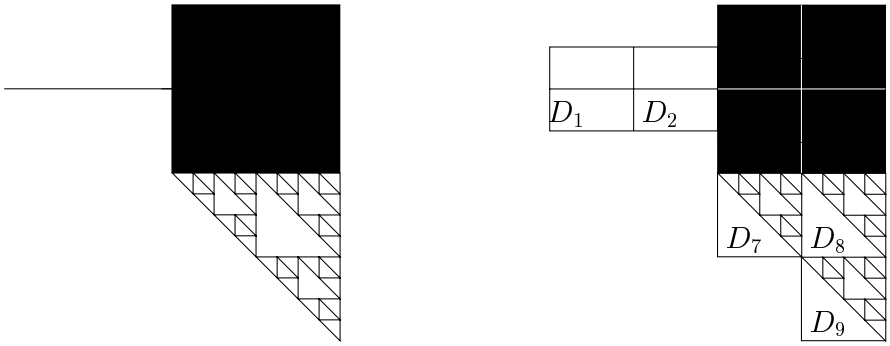
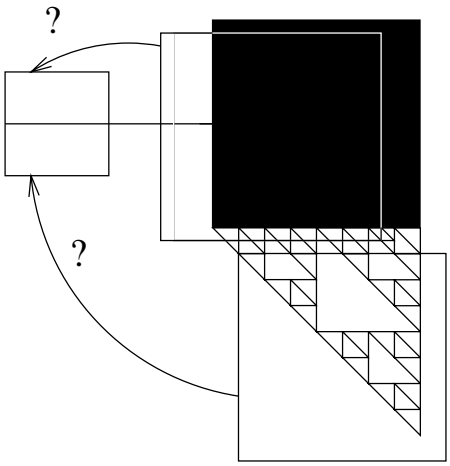
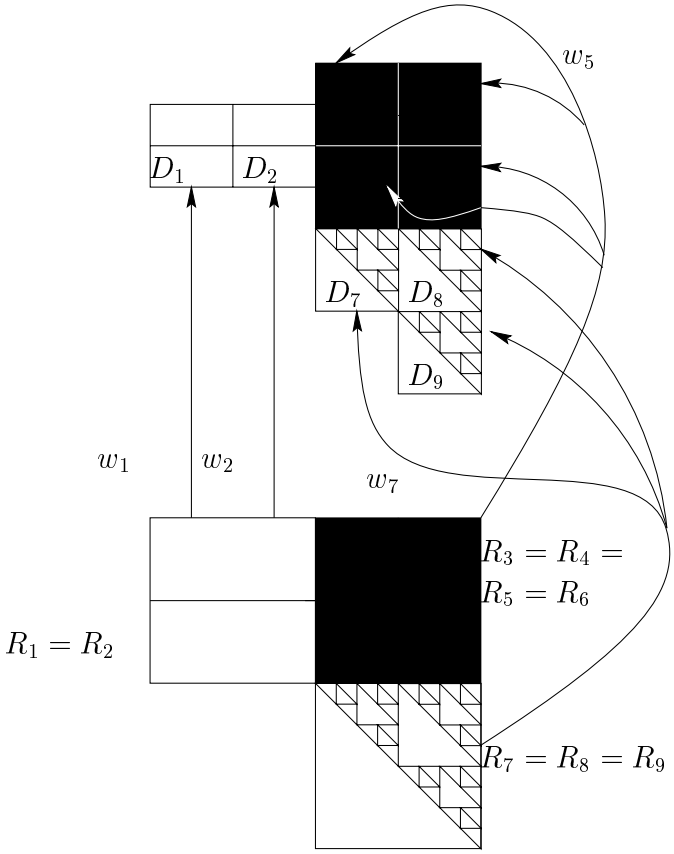


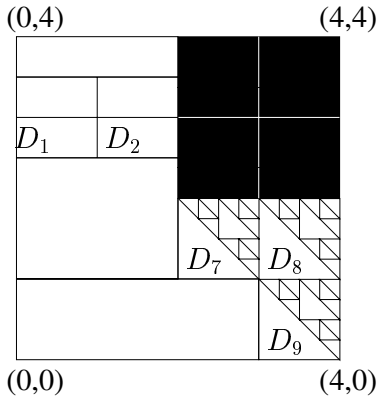
Bild G und eine Überdeckung mit Domainblöcken



Suche nach geeigneten Range-Blöcken



Zuordnung von Range- und Domainblöcken



Koordiantensystem zur Referenz einer linken unteren Reckteck-Ecke

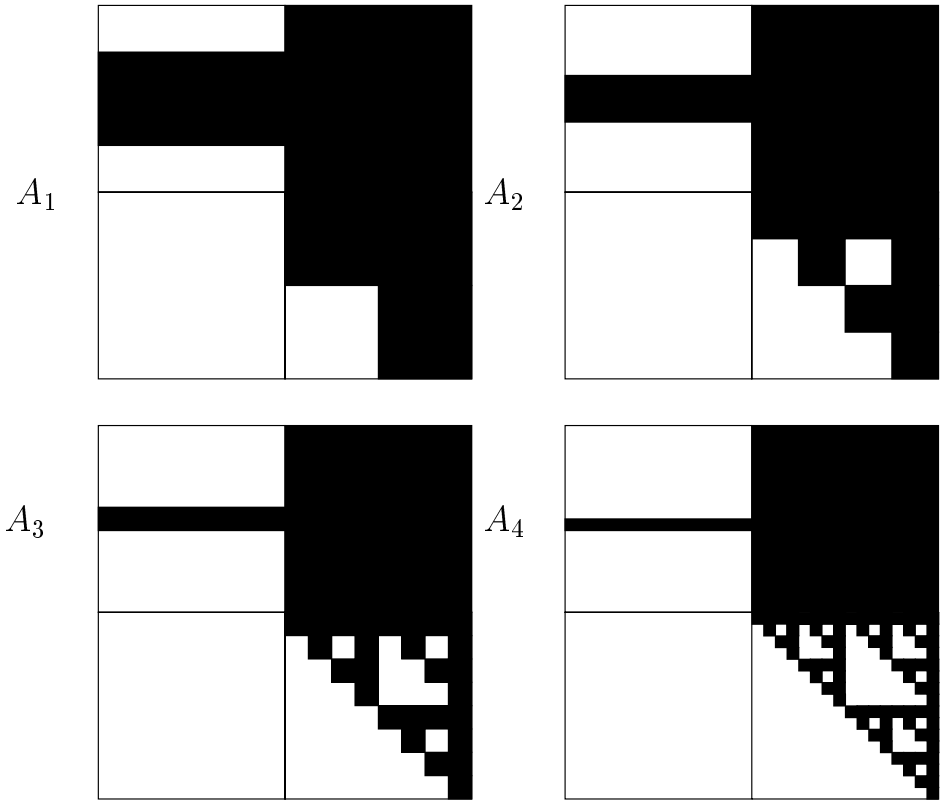
Map#	D_x	D_y	R_x	R_y	Symmetrie
1	0	2.5	0	2	0
2	1	2.5	0	2	0
3	2	2	2	2	0
4	3	2	2	2	0
5	2	3	2	2	0
6	3	3	2	2	0
7	2	1	2	0	0
8	3	1	2	0	0
9	3	0	2	0	0

Beschreibung eines Schwarzweißbildes durch 9 affine Abbildungen
von Range-Blöcken mit Koordinanten R_x, R_y
auf Domain-Blöcke mit Koordinanten D_x, D_y.

Eine einfache Methode zur Bestimmung des Attraktors A eines IFS (iterierten Funktionensystems) ist der Escape-Time-Algorithmus:

Da alle Domain-Blöcke $D_1, \ldots, D_n$ eine Überdeckung der schwarzen Bildteile von R ergeben, hat jedes $x \in D = D_1 \bigcup D_2 \bigcup \ldots \bigcup D_n$ durch die für seine Domain zuständige Abbildung $w_i : R_i \to D_i$ ein eindeutiges Urbild $f(x) := w_i^{-1}(x)$.
Sei eine maximale Iterationszahl `MAX_ITER` vorgegeben.

```
Färbe alle Pixel des (vorläufigen) Attraktors A schwarz.
Für jedes Pixel x0 tue:
x = x0
for (z = 0; ((z++ < MAX_ITER) && (x ∈ D)); x = f(x));
if (x ∉ D) färbe x weiß (d.h. gehört nicht zum Attraktor)
```



Auswirkungen der ersten 4 Iterationen des Escape-Time-Algorithmus

Bei Grauwertbildern wird der Helligkeitswert als 3. Dimension geführt, und die affinen Abbildungen haben die Form:

$$w_i = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & P \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} R_x \\ R_y \\ Q_i \end{bmatrix} ,$$

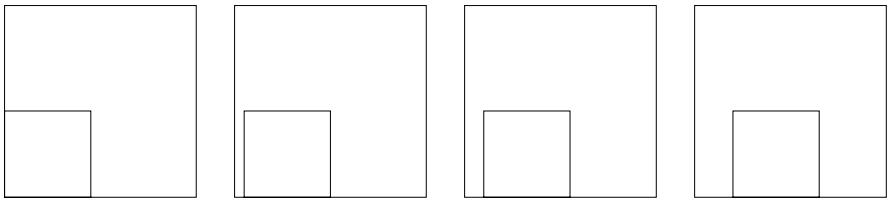
wobei die Koeffizienten a_i, b_i, c_i, d_i Transformationen beschreiben gemäß der 8 möglichen Symmetrien mit einem Kontraktivitätsfaktor $s = \frac{1}{2}$. P ist eine feste Konstante unterhalb des Kontraktivitätsfaktors s, Q_i eine individuelle Helligkeitsverschiebung. Der Grauwert z wird daher abgebildet auf

$$v_i(z) = P \cdot z + Q_i .$$

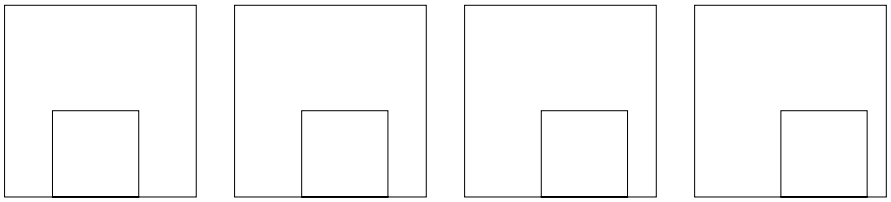
Die Domain-Blöcke D_i bilden nun eine Partition des Rechtecks R; die Range-Blöcke mit doppelter Kantenlänge beginnen mit ihrer linken unteren Ecke an endlich vielen, dafür vorgesehenen Gitterpunkten. Zur Bestimmung des Attraktors werden, ausgehend von einem beliebigen Rechteckinhalt, immer wieder die lokalen Abbildungen w_i auf ihre Range-Blöcke R_i angewendet und berechnen ein Update der zugehörigen Domain-Blöcke.

13	14	15	16
9	10	11	12
5	6	7	8
1	2	3	4

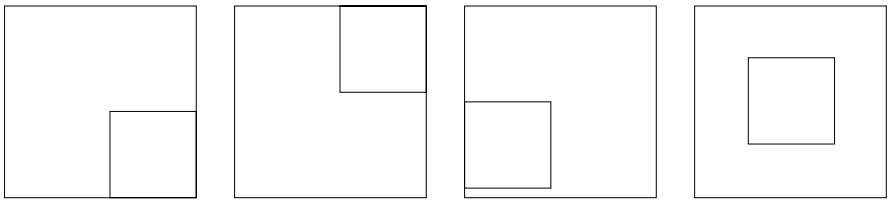
Partitionierung des Bildes in Domain-Blöcke



$R_x = 0, R_y = 0$ $R_x = 1, R_y = 0$ $R_x = 2, R_y = 0$ $R_x = 3, R_y = 0$



$R_x = 4, R_y = 0$ $R_x = 5, R_y = 0$ $R_x = 6, R_y = 0$ $R_x = 7, R_y = 0$

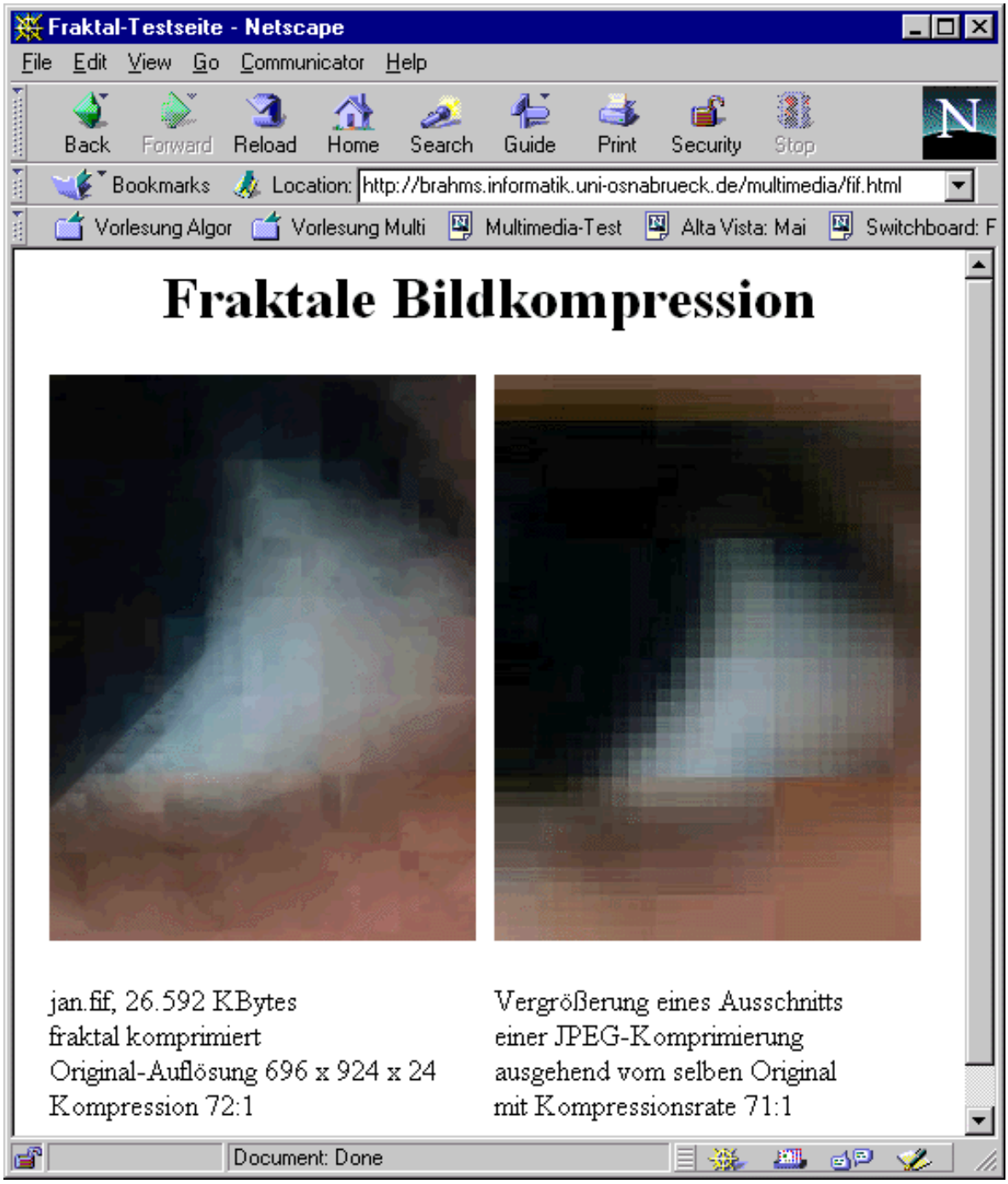


$R_x = 8, R_y = 0$ $R_x = 8, R_y = 8$ $R_x = 0, R_y = 1$ $R_x = 4, R_y = 4$

einige Anfangspositionen der Range-Blöcke

Map	R_x	R_y	Symmetrie	Q
1	0	8	0	-7
2	0	8	6	29
3	0	4	4	50
4	0	8	0	-4
5	0	8	0	-7
6	3	4	2	50
7	3	5	5	-16
8	0	6	3	50
9	0	8	0	-7
10	2	5	0	58
11	0	8	2	48
12	0	8	1	32
13	0	8	0	-7
14	0	5	5	-12
15	0	8	5	23
16	0	8	0	-7

Beispiel für fraktale Beschreibung eines Grauwertbildes durch 16 affine Abbildungen



Screenshot vom Fraktal-Plugin

Kapitel 6

Bilddateiformate

6.1 TIF

Eines der häufig verwendeten Dateiformate zur Speicherung von Pixelbildern wurde von Aldus Corporation, Seattle, USA, entwickelt: *Tag Image File Format*, abgekürzt TIF.

Dateikopf:	1. Wort	`4d4d` Motorola (high order first)
		`4949` Intel (low order first)
	2. Wort	`002a` Versionsnummer (konstant)
	3. + 4. Wort	Offset des ersten IFD (Image File Directory)
IFD:	1. Byte	Anzahl der Tags im IFD, jedes Tag besteht aus 12 Bytes
	Aufbau eines Tag:	
	1. Wort	Identifikation des Tags (es gibt 45 Tags)
	2. Wort	Typ der Daten
		`1`: Byte
		`2`: 0-terminierter String von ASCII-Zeichen
		`3`: short = 16-Bit unsigned integer `(0000-ffff)`
		`4`: long = 32-Bit unsigned integer `(00000000-ffffffff)`
		`5`: Rational = Bruch aus 2 long-Werten
		`11`: float = im IEEE-Format, einfache Genauigkeit
		`12`: double = im IEEE-Format, doppelte Genauigkeit
	3. + 4. Wort	Anzahl der Daten für dieses Tag
	5. + 6. Wort	Tag-Daten falls mit ≤ 4 Bytes darstellbar : von links nach rechts füllen falls mehr als 4 Bytes benötigt: 32-Bit Startadresse für Daten
Datenblock:		RGB-Werte oder Indizes (ggf. komprimiert) für die Farbtabelle

Offset	Bedeutung	Wert
0100	ImageWidth	z.B. 814
0101	ImageLength	z.B. 517
0102	Bits per Sample	z.B. 8, ggf. Zeiger auf 3 shorts
0103	Compression	1 keine Komprimierung 2 CCITT Gruppe 3 (nur bei Schwarz/Weiß) 3 CCITT Gruppe T4 (nur bei Schwarz-Weiß) 4 CCITT Gruppe T6 (nur bei Schwarz/Weiß) 5 LZW 6 JPEG 32773 Packbit mit Lauflängen-Kodierung
0106	Photometric Interpretation	0 S/W mit 0=Weiß, wachsende Nummern gegen Schwarz 1 S/W mit 0=Schwarz, wachsende Nummern gegen Weiß 2 RGB-Farbbild, pro Pixel drei Farbwerte, 0,0,0=Schwarz 3 Palettenbild, pro Pixel ein Index in Palette 4 Transparenzmaske für ein weiteres Bild 5 CMYK-Farbsystem, pro Pixel 4 Farbwerte 6 YUV-Farbsystem, pro Pixel Luminanz + 2 Chrominanz
0111	Strip Offset	Startadresse des Datenblocks (meistens $0008)
0112	Orientation	1 horizontal, beginnend oben links
0115	Samples per Pixel	3 bei RGB-Bildern 1 sonst
0116	Rows per Strip	wenn 1 Block: Bildhöhe wenn > 1 Block: Anzahl Zeilen pro Block
0117	Strip Byte Counts	wenn 1 Block: Anzahl der Bytes im Block wenn > 1 Block: Zeiger auf Feld mit Streifenlängen
011a	X Resolution	Zeiger auf 2 long Werte, zB. 300 1
011b	Y Resolution	Zeiger auf 2 long Werte, zB. 300 1
0128	Resolution unit	1 keine Einheit 2 Inch 3 Zentimeter
011c	Planar Configuration	1 RGBRGBRGB ... 2 RRRR... GGGG.... BBBB....
0140	Color map	Anzahl der Einträge + Startadresse Jeder Farbwert $0 \leq w = 255$ wird als Wort ww abgelegt

Tag-Art : True Color Bild

						Position 0:
4d4d						Motorola
002a						Versionsnr. (konstant)
0013	43ba					Offset des IFD $001343ba = 1262522
						Position 8:
ebe9	e7e2	e4c0	b3bd	9a9c	a387	Beginn der RGB-Werte, insgesamt 814*517*3 Bytes
9178	5294	724b	8d6f	4b8e	7453	
a79e	6da8	a06f	a19b	759d	a486	
a6ac	8aa2	aa92	a7ac	9ba5	a995	
...						
...						
...						
						Position 1262522: Beginn des Image File Directory
000b						$000b = 11 Tags zu je 12 Bytes
0100	0003	0000	0001	032e	0000	Breite, 1 short, Wert 3*256 + 2*16 + 14 = 814
0101	0003	0000	0001	0205	0000	Höhe, 1 short, Wert 2*256 + 0*16 + 5 = 517
0102	0003	0000	0003	0013	4444	Bits per sample, 3 short, Adresse $134444=1262660
0103	0003	0000	0001	0001	0000	Compression, 1 short, Wert=1 (keine Kompression)
0106	0003	0000	0001	0002	0000	Photometric, 1 short, Wert=2 (RGB-Farbbild)
0111	0004	0000	0001	0000	0008	StripOffsets,1 long, Wert=8 (Startadresse)
0112	0003	0000	0001	0001	0000	Orientierung,1 short, Wert = 1 (zeilenweise links/oben)
0115	0003	0000	0001	0003	0000	Samples per Pixel, 1 short, Wert=3 (RGB)
0116	0004	0000	0001	0000	0205	Rows per Strip, 1 long, Wert $205 = 517 (Zeilen)
0117	0004	0000	0001	0013	43b2	StripByteCounts, 1 long, Wert = 1262514 (Bytes)
011c	0003	0000	0001	0001	0000	Planar Konfiguration, 1 short, Wert=1 (RGBRGB...)
0000	0000					Ende IFD, kein weiteres IFD
						Position 1262660:
0008	0008	0008				jeweils 8 Bit pro Farbwert

kommentierter Hex-Dump zu einem True-Color-Bild im tif-Format

Vorspann	8 Bytes
814 * 517 RGB-Tripel	1262514 Bytes
IFD 11 * 12 + 12	144 Bytes
→ Dateilänge	1262666 Bytes

Tag-Art : Palettenbild

						Position 0:
4d4d						Motorola
002a						Versionsnr. (konstant)
0006	6be					Offset des IFD $00066bee = 420846
						Position 8:
fe68	bd52	6f6f	6f6f	0101	01f4	Beginn der Indizes, insgesamt 814*517 Bytes
5252	e952	2abd	522a	5252	b252	
0000	016f	076f	01bd	e952	3877	
ea3a	3838	38a1	0101	3a38	01be	
.						
.						
						Position 420846: Beginn des Image File Directory
000c						$000c = 12 Tags zu je 12 Bytes
0100	0003	0000	0001	032e	0000	Breite, 1 short, Wert $32e = 814
0101	0003	0000	0001	0205	0000	Höhe, 1 short, Wert $205 = 517
0102	0003	0000	0001	0008	0000	Bits per sample, 1 short, Wert = 8 (Adresse)
0103	0003	0000	0001	0001	0000	Compression, 1 short, Wert=1 (keine Kompression)
0106	0003	0000	0001	0003	0000	Photometric, 1 short, Wert=3 (Palette)
0111	0004	0000	0001	0000	0008	StripOffsets,1 long, Wert=8 (Startadresse)
0112	0003	0000	0001	0001	0000	Orientierung,1 short, Wert=1 (zeilenweise l.o.)
0115	0003	0000	0001	0001	0000	Samples per Pixel, 1 short, Wert=1 (Palette)
0116	0004	0000	0001	0000	0205	Rows per Strip, 1 long, Wert $205 = 517 (Zeilen)
0117	0004	0000	0001	0006	6be6	StripByteCounts,1 long, Wert = 420838 (Bytes)
011c	0003	0000	0001	0001	0000	Planar Konfiguration, 1 short, Wert=1 (RGBRGB..)
0140	0003	0000	0300	0006	6c84	Colormap, $300 = 768 short, beginnend bei $66c84
0000	0000					Ende IFD, kein weiteres IFD
						Position: $66c84 = 420996
						Palette mit 768 Doppelbytes
b8b8	a8a8	1010	3c3c	0808	2424	
b4b4	a8a8	1c1c	9898	6868	3838	
3c3c	5858	2c2c	7070	6464	8080	
.						
.						
3030	1414	1010	3030	2020	3c3c	
7070	2828	d8d8	1010	2828	2020	
7474	1414	acac	a8a8	e4e4	e4e4	

kommentierter Hex-Dump zu einem Palettenbild im tif-Format

Vorspann	8 Bytes
814 * 517 Indizes	420838 Bytes
IFD 12 * 12 + 6	150 Bytes
Palette 3 * 256 * 2	1536 Bytes
→ Dateilänge	422532 Bytes

6.2 Photo-CD

Die von Kodak entwickelte Photo-CD weist eine Verzeichnisstruktur auf, die zum Ansteuern eines Photo-CD-Players ausgelegt ist und außerdem von diversen Bildverarbeitungsprogrammen gelesen werden kann. Hierfür stellt Kodak eine Programmierbibliothek zur Verfügung; das Erstellen einer Photo-CD ist nur speziellen Labors vorbehalten. Typischer Preis: 16,00 DM für CD, 5,00 DM für Session, 1,20 DM pro Bild. Es können 100 Photos, ggf. in mehreren Sessions, aufgebracht werden. Da sich dadurch das Verzeichnis an verschiedenen Stellen der CD befinden kann, muß das CD-Laufwerk "multisessionfähig" sein.

Jedes Photo wird in 5 verschiedenen Auflösungen in einem (patentrechtlich geschützten) Image-Pac abgelegt.

Bezeichnung	Auflösung	unkomprimierter Platzbedarf
Base/16	192×128	72 KBytes
Base/4	384×256	288 KBytes
Base	768×512	1152 KBytes
Base$*$4	1536×1024	4608 KBytes
Base$*$16	3072×2048	18432 KBytes

Beim Einscannen werden die RGB-Werte für jeden Bildpunkt mit 12 Bit Genauigkeit pro Farbanteil abgetastet und anschließend in das Kodak-eigene YCC-Format konvertiert (8 Bit für Helligkeit, je 8 Bit für zwei Chrominanzwerte). Durch 4:1:1 Subsampling wird die Chrominanz über je 4 Pixel gemittelt. Pro Image-Pac werden die ersten 3 Auflösungen explizit gespeichert, für Base$*$4 und Base$*$16 nur die Differenzen. Dadurch reichen etwa 4.5 MByte aus, also etwa 20 % des aufsummierten Platzbedarfs.

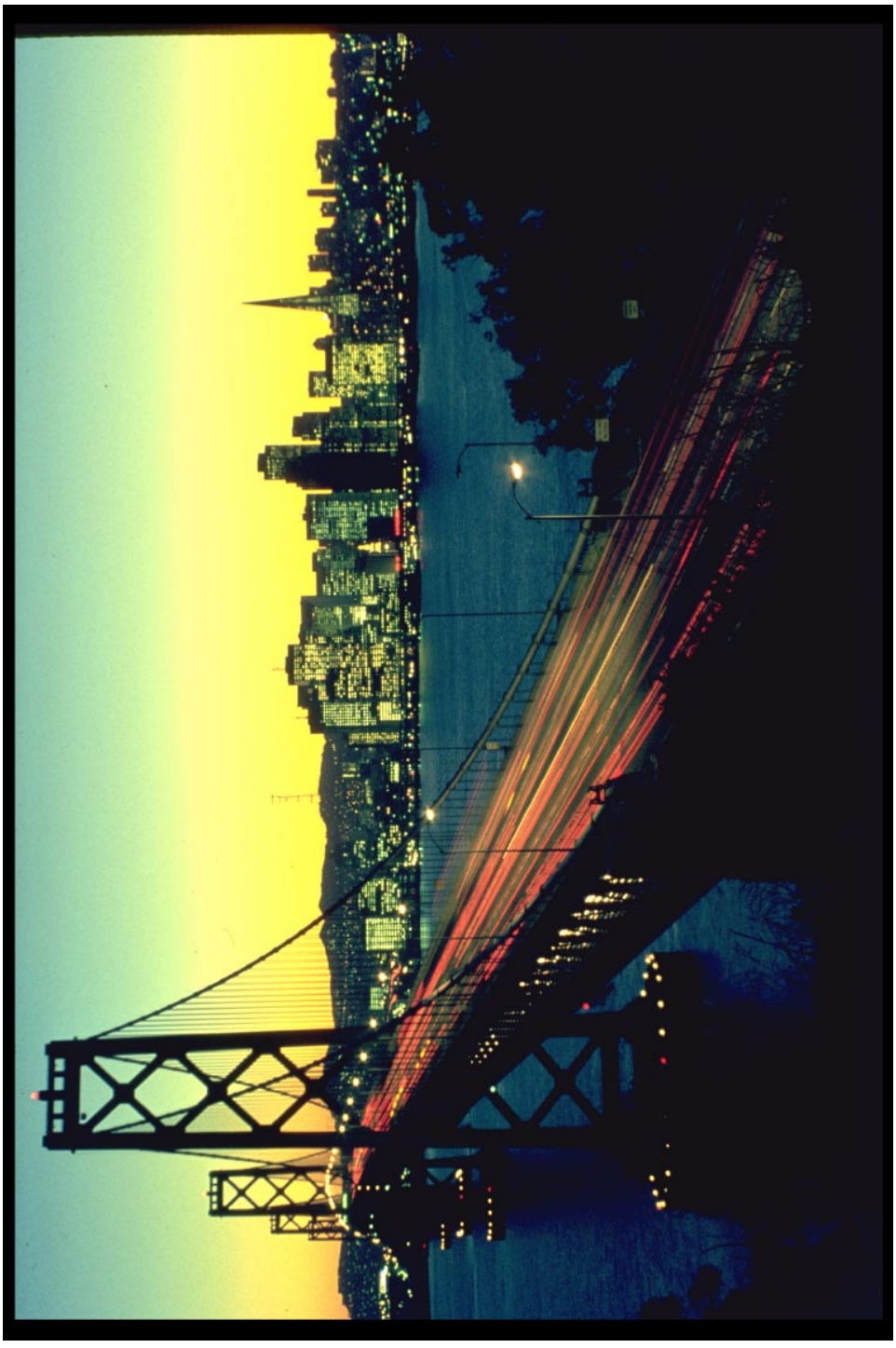
6.3 Auflösung

Für die Qualität eines Bildes sind die am Erstellungsprozeß beteiligten Auflösungen verantwortlich. Die Auflösung wird meistens gemessen als *Dots per Inch* (dpi).

Scanner-Auflösung:	Gegeben durch Anzahl der CCD-Elemente in einer Sensorzeile (z.B. 300 dpi).
Scan-Auflösung:	Gewählte Auflösung beim Scanner.
Bildauflösung:	Entspricht zunächst der Scan-Auflösung, kann später ggf. runtergerechnet werden (durch Mittelung) oder hochgerechnet (durch Interpolation).
Bildschirmauflösung:	Ein 17-Zoll Monitor mit Seiten-Verhältnis 4:3 hat eine Breite von $(17 \cdot 4)/5 = 13.6$ inch. Bei 1024×768 Pixeln ergibt das $1024/13.6 \approx 75$ dpi.
Druckerauflösung:	Gegeben durch die Anzahl der Druckerpunkte, die der Druckkopf nebeneinander setzen kann (z.B. 300 dpi). Da bei Tintenstrahldruckern nur 3 Grundfarben (ggf. zusätzlich Schwarz) zur Verfügung stehen, wird jedes Farbpixel durch eine Rasterzelle mit 16×16 Druckerpunkten dargestellt.
Druckauflösung:	Druckerauflösung/Rasterzellengröße, d.h. ein 400 dpi-Belichter mit 16×16 Rasterzellen kann nur $400/16 = 25$ dpi erzeugen.

Thermosublimationsdrucker lösen durch Heizelemente Farbpigmente aus einer Folie, die anschließend in ein Spezialpapier eindringen. Jedes True-Color-Pixel wird daher als ein Farbpunkt gedruckt (*Continuous Tone-Druck*). Ein 300 dpi Thermosublimationsdrucker ermöglicht daher eine 300 dpi Druckauflösung. Ein 100 ASA-Kleinbilddia, welches eine reale Auflösung von 2000 dpi in sich birgt, kann daher mit einem 300 dpi-Thermosublimationsdrucker in idealer Weise auf das $2000/300 = 6.66$-fache vergrößert werden, welches eine Kantenlänge von $6.66 * 36$ mm $= 24$ cm bedeutet (etwa DIN-A4). Allerdings ist der im analogen Dia enthaltene Helligkeitsumfang etwa um den Faktor 10 größer als ein Farbdruck oder Monitorbild darstellen kann.

Auflösung KB-Dia
Die Projektion eines 24×36 mm Kleinbilddias auf eine 1.80 m breite Leinwand stellt eine 50-fache Vergrößerung dar. Sind dann auf 1 cm Leinwand 8 Linien zu unterscheiden, entspricht das 16 dots per cm Leinwand= 800 dots per cm Dia $\approx$ 2000 dpi.

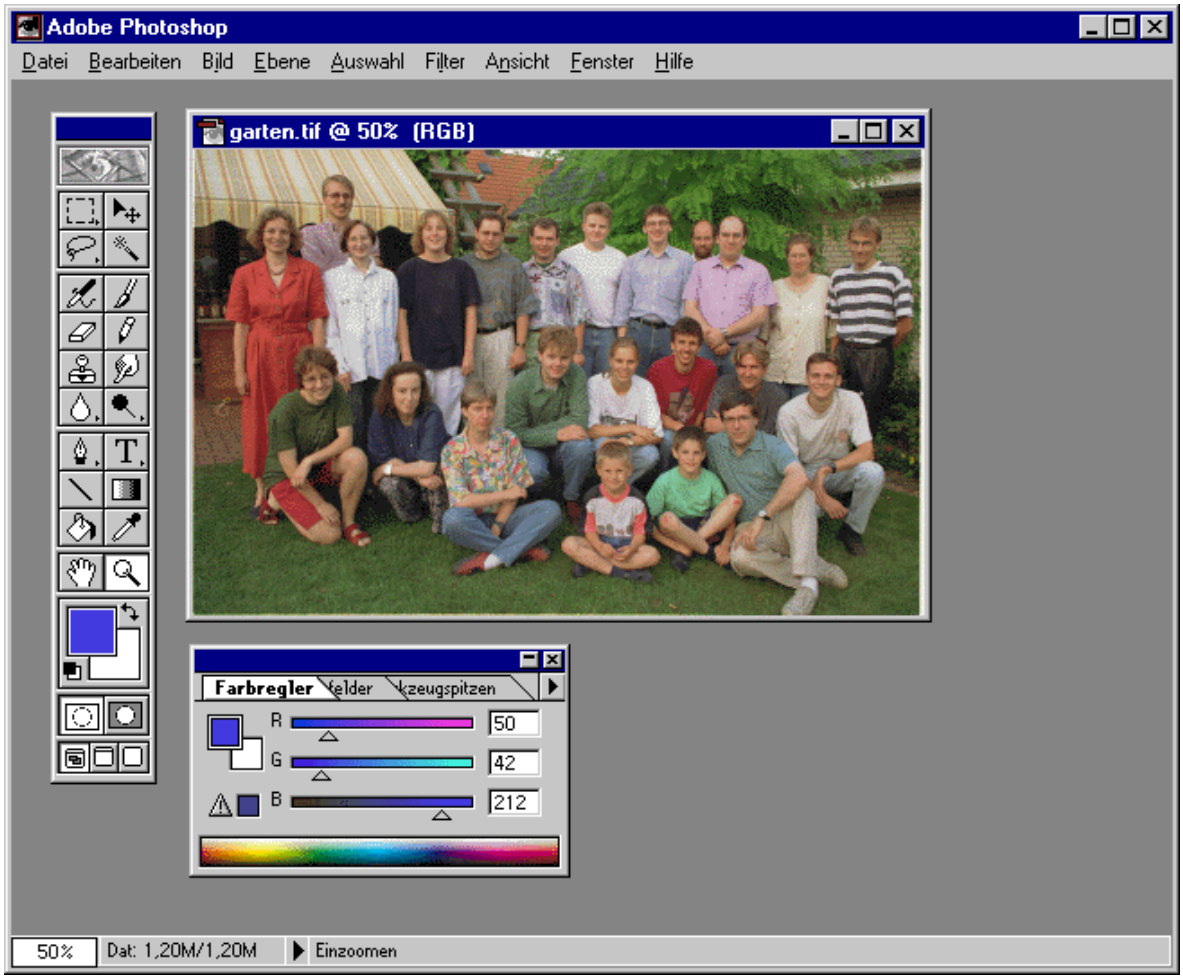


24 x 36 mm Dia, eingescannt mit 2168 dpi, ergibt 3072 x 2048 Pixel
skaliert auf 75 %, ergibt 2304 x 1536 Pixel
gedruckt mit 300 dpi Ink Jet, ergibt 19.5 x 13.0 cm

6.4 Adobe Photoshop

Adobe Photoshop ist ein Programm zum Erstellen von 2D-Grafiken und zur Nachbearbeitung von Pixel-Dateien. Die charakteristischen Leistungsmerkmale lauten:

- Erzeugen und Manipulieren von 2D-Objekten wie Linien, Kreise, Polygone mit diversen Malwerkzeugen,
- Importieren von Pixel-Dateien verschiedener Formate,
- vielfältige Selektionsmöglichkeiten der zu bearbeitenden Bildteile,
- geometrische Transformationen und Verzerrungen,
- Verpflanzung von Bildteilen durch Stempelwerkzeug,
- Filteroperationen,
- Kantendetektion,
- Weich- und Scharfzeichner,
- Kontrastmanipulation,
- Retusche durch Aufhellen, Abdunkeln, "Verschmieren",
- Wechsel der Farbtiefe,
- Dithering,
- Konvertierung der Bildformate,
- JPEG-Kompression.



Screenshot vom Bildbearbeitungswerkzeug Adobe Photoshop

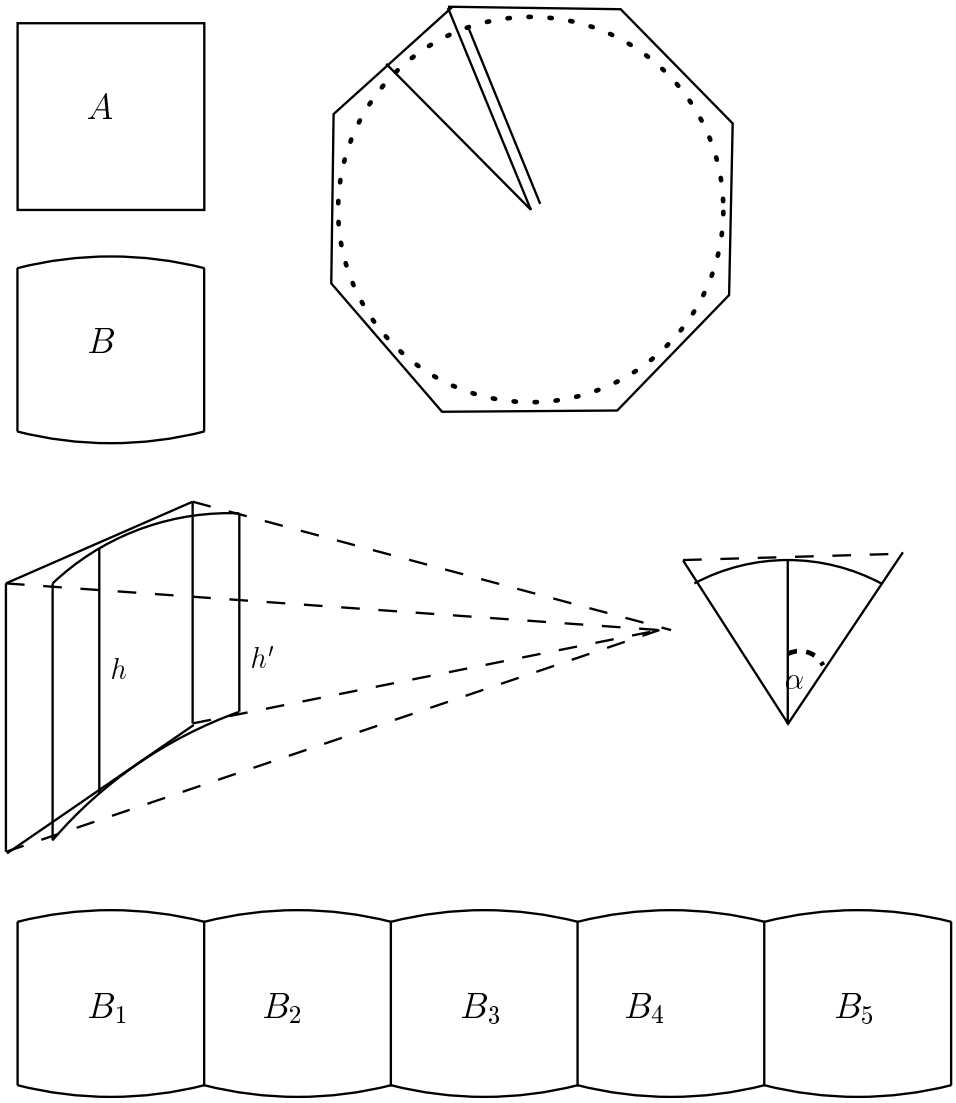
6.5 Apple Quicktime Virtual Realilty

Apple Quicktime Virtual Reality Bilder beinhalten 360° Rundblicke, abgespeichert in einem Format, welches mithilfe des entsprechenden Browser-Plugins ein interaktives Betrachten erlaubt.
Zunächst werden mit einer Kamera auf Stativ etwa 12-16 Einzelaufnahmen im Kreis gemacht. Nach perspektivischer Stauchung jedes Einzelbildes an den Rändern werden alle Einzelbilder zu einem Panorama verschmolzen. Beim Betrachten im Browser wird jeweils der angeschaute Bereich in Realzeit zurücktransformiert, so daß der Eindruck eines wechselnden Gesichtsfeldes entsteht.

5 Einzelbilder

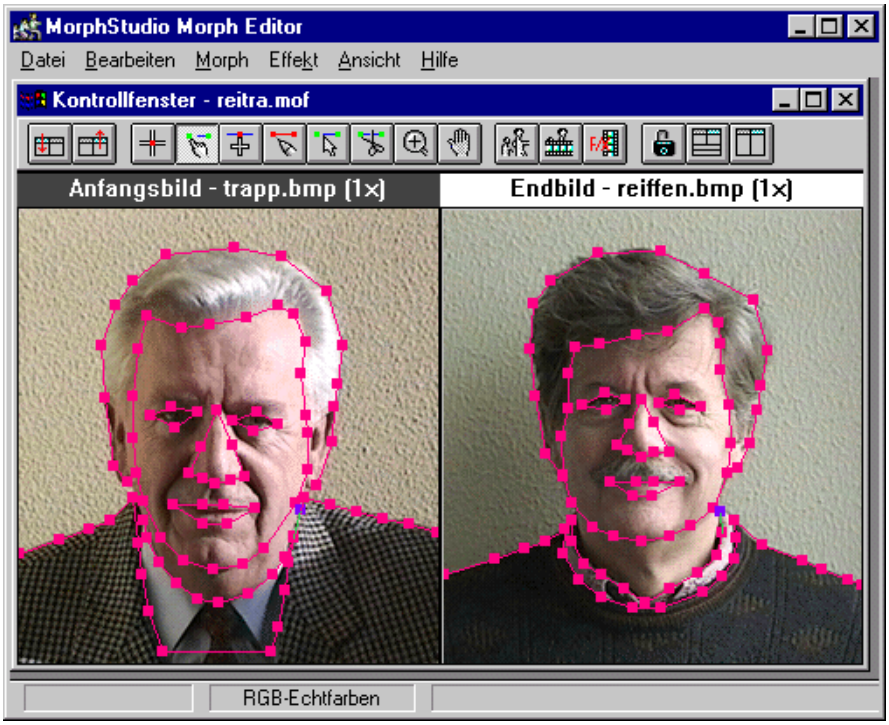
verschmolzene Einzelbilder

Der zum Stauchen benötigte Faktor h'/h verkürzt jede vertikale Linie eines Rechtecks A in Abhängigkeit vom Winkel α. Es entsteht ein "Kissen" B. Nach dem Aneinanderreihen werden die überlappenden Teile zur Deckung gebracht und die herausragenden Buckel oben und unten abgeschnitten.



6.6 Morphing

Unter *Morphing* versteht man die gleichförmige Transformation eines gegebenen Quell-Bildes in ein gegebenes Ziel-Bild. Hierzu werden in Quelle und Ziel korrespondierende Punkte und Kanten plaziert, die als Vorgabe für die geometrische Transformation dienen. Im Kräftefeld aller dieser Abbildungen wandert jedes Pixel der Quelldatei zu seinem korrespondierenden Zielpixel unter gleichzeitiger Farbanpassung.

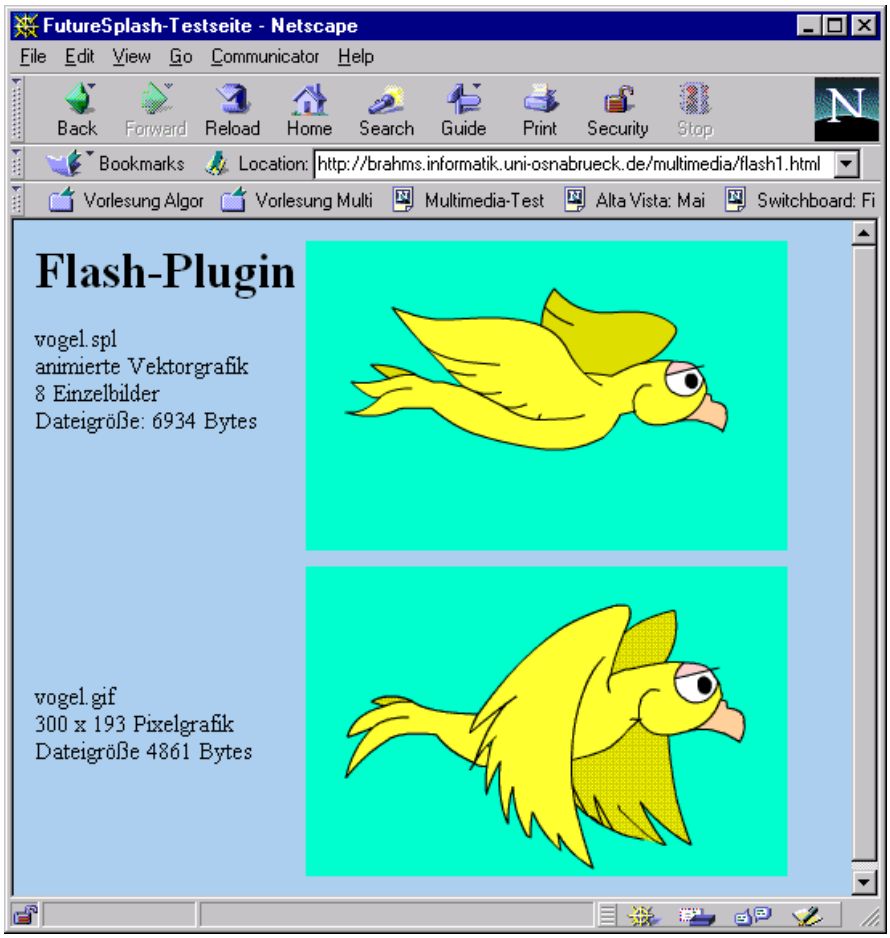


Screenshot vom Morph-Editor Ulead Morph Studio

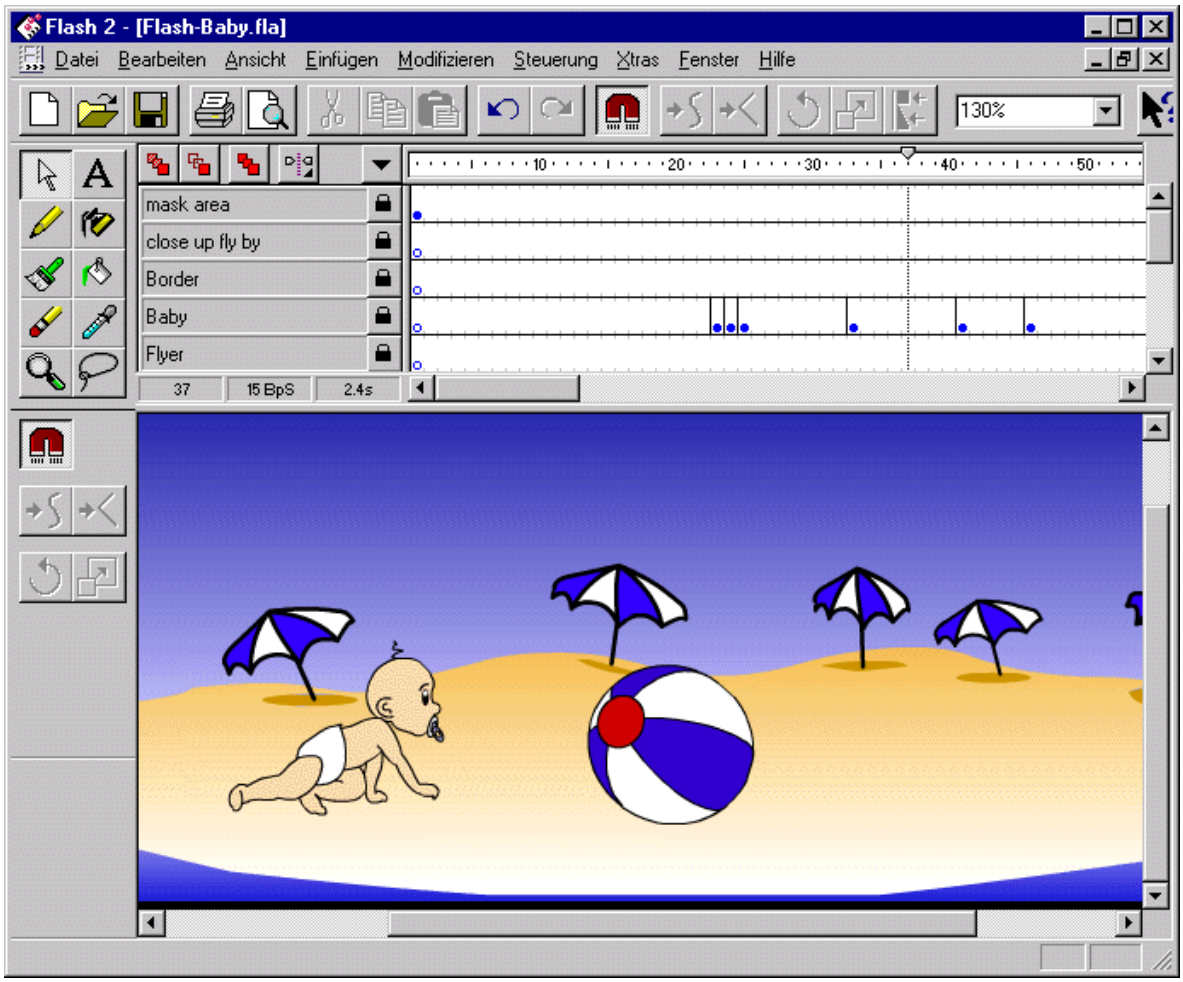
Morphresultat: Professor Treiffen

6.7 Macromedia Flash

Macromedia Flash ist ein Werkzeug zum Editieren und Animieren von zweidimensionalen Vektorgrafiken. Durch sein kompaktes Speicherformat eignet es sich besonders zur Herstellung plakativer, animierter Grafiken zur Internetpräsentation.



Screenshot vom Flash-Plugin



Screenshot vom Vektorgrafikwerkzeug Macromedia Flash

Kapitel 7

Computergrafik

Unter Computergrafik versteht man die Generierung einer photorealistischen Darstellung anhand der Beschreibung einer 3-dimensionalen Szene inkl. Beleuchtung und synthetischer Kamera. Typische Einsatzgebiete sind

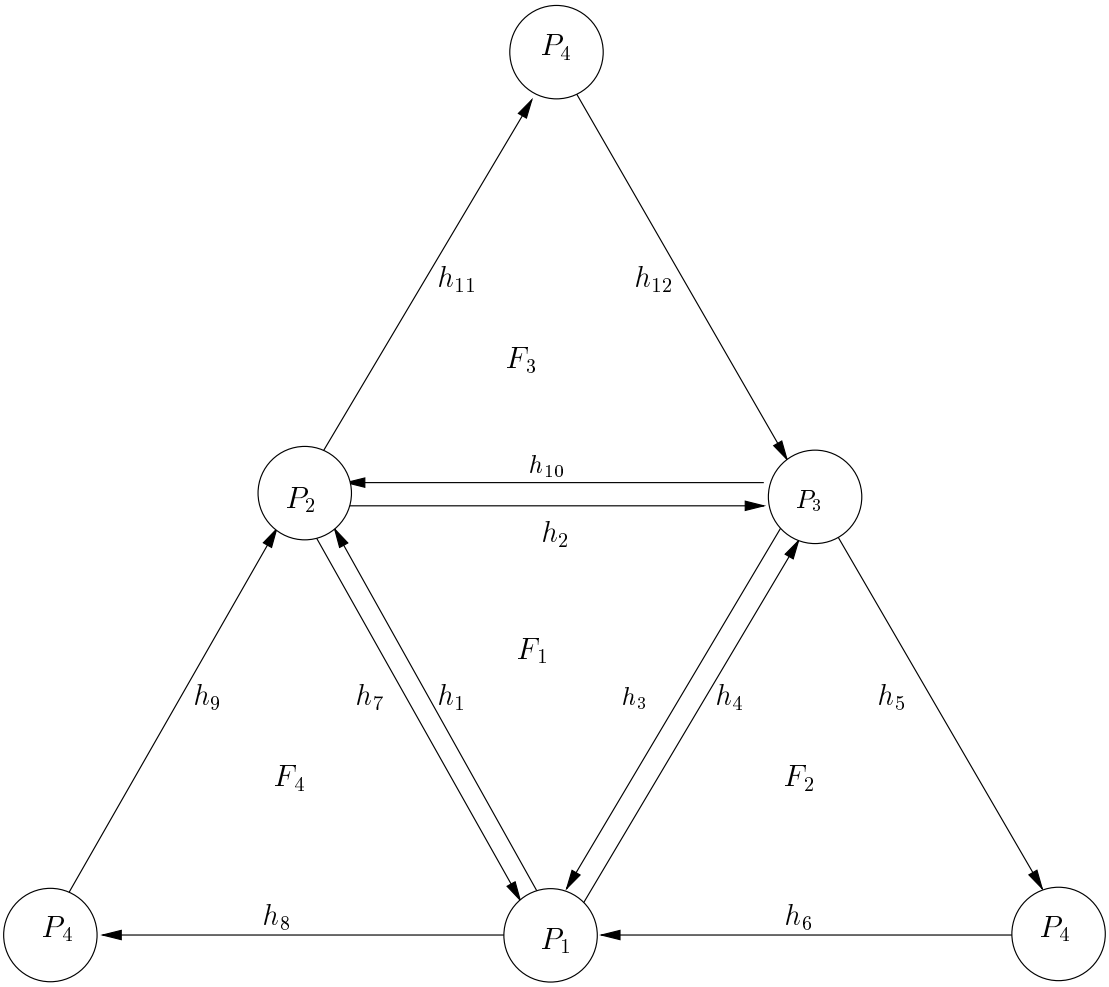
- CAD (Architektur & Maschinenbau)
- Visualisierung (von Meßergebnissen)
- Simulation (von physikalischen/chemischen/biologischen Vorgängen)
- Unterhaltung (Virtual Reality)

Zur interaktiven Manipulation der Szene ist ein 3D-Editor erforderlich, der auf einer geeigneten Repräsentation zur Verwaltung der spezifizierten Objekte operiert. Mit Hilfe des Editors werden 3D-Transformationen auf selektierte Objekte angewendet. Der Betrachterstandpunkt wird durch eine synthetische Kamera definiert. Zur Berechnung der Projektion durchlaufen alle Objekte eine Viewing Pipeline. Unter Berücksichtigung von Materialeigenschaften und Lichtquellen werden die projizierten Flächen eingefärbt. Nicht sichtbare Pixel werden mit Hilfe eines Tiefenpuffers entdeckt. Lichtbrechung und Lichtreflexion können durch *Ray tracing* simuliert werden.

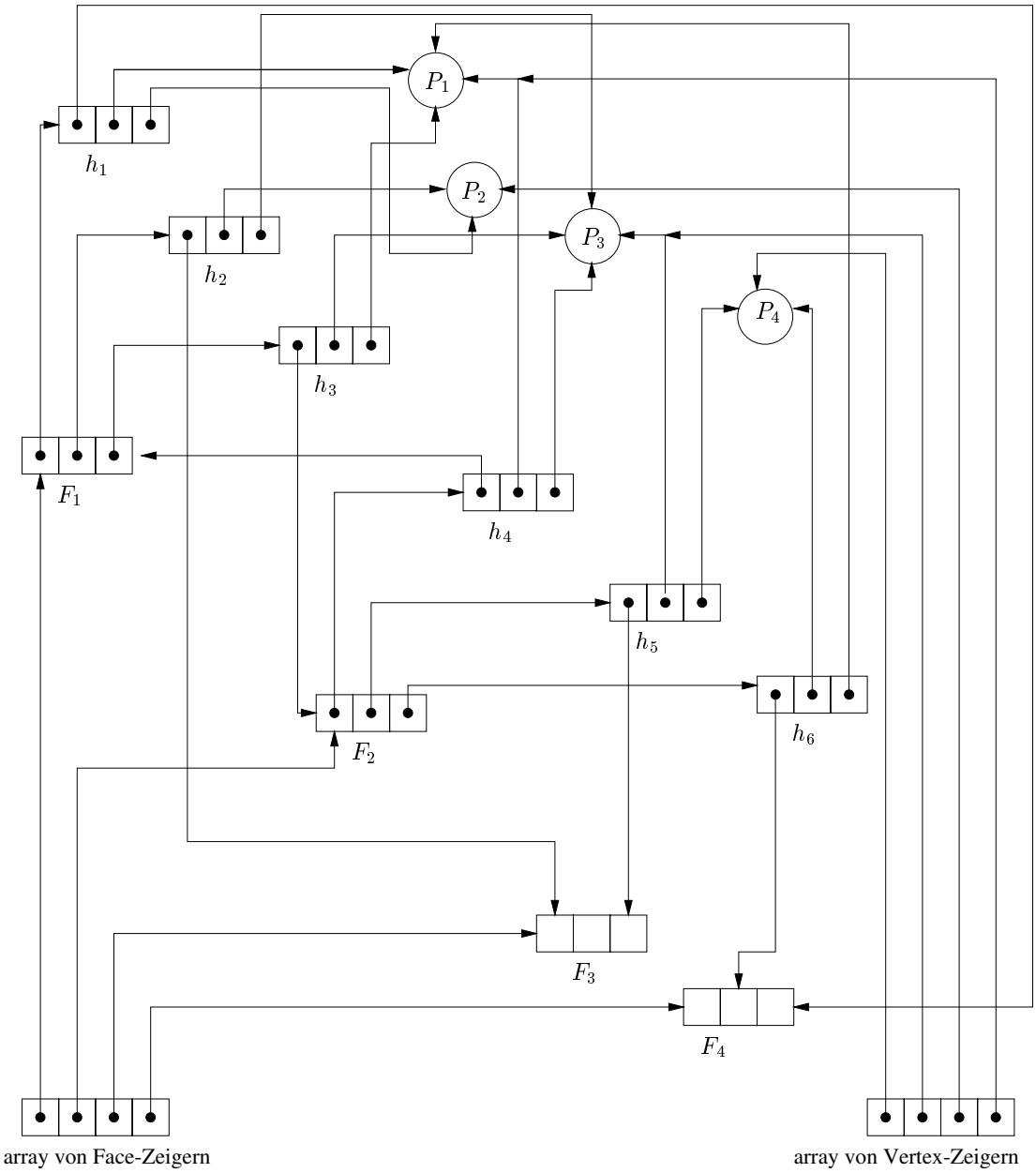
3-D-Editor
Ein 3-D-Editor stellt am Bildschirm ein 3-dimensionales Objekt als Drahtgittermodell dar, und zwar perspektivisch oder als Grund-, Auf-, Seitenriß. Es können Punkte, Kanten und Flächen selektiert und transformiert werden durch Translation, Skalierung und Rotation.

7.1 Repräsentation

Viele Gegenstände lassen sich durch Polyeder approximieren. Eine hierfür geeignete Datenstruktur ist die Halbkantenrepräsentation: Jedes Objekt besteht aus einer Liste von Polygonen. Jedes Polygon besteht aus einer Liste von Halbkanten mit Verweisen auf die beiden Endpunkte und die benachbarte Polygonfläche sowie einer Flächennormalen und ggf. Oberflächenbeschaffenheit. Alle Transformationen werden nur auf den Endpunkten durchgeführt; die Normale hilft bei der Bestimmung verdeckter Flächen, die Oberflächenbeschaffenheit beeinflußt zusammen mit den Lichtquellen das Rendern und Raytracing.



Tetraeder mit 4 Knoten, 12 Halbknoten, 4 Flächen



Verzeigerungsstruktur für Tetraeder. Nicht gezeichnet sind die Halbkanten der Flächen F_3 und F_4.

7.2 2D-Grafik

Zum Zeichnen des projizierten Drahtgittermodells und zur Visualisierung der an einem Objekt beteiligten Polygonflächen wird eine Routine zum Generieren einer Geraden benötigt, die für zwei gegebene 2D-Punkte ausrechnet, welche Pixel längs der Verbindungslinie anzuschalten sind (Bresenham-Algorithmus).
Zum Zeichnen einer Kurve durch n vorgegebene Punkte werden $n-1$ kubische Polynome (*Splines*) bestimmt, die an den Nahtstellen jeweils identische erste und zweite Ableitungen besitzen. Anschließend wird durch die Wahl von geeignet vielen Interpolationspunkten auf den Kurvenabschnitten der Gesamtverlauf durch Geradenstücke approximiert.

7.3 3D-Transformation

Ein Punkt mit den Koordinaten (x, y, z) wird bewegt an die Stelle (x', y', z'):

Translation	$x' := x + t_x$ $y' := y + t_y$ $z' := z + t_z$
Skalierung bzgl. Nullpunkt	$x' := x \cdot s_x$ $y' := y \cdot s_y$ $z' := z \cdot s_z$
Skalierung bzgl. Punkt P	Translation um $-P$ Skalierung Translation um $+P$
Rotation um z-Achse	$x' := x \cdot \cos(\alpha) - y \cdot \sin(\alpha)$ $y' := x \cdot \sin(\alpha) + y \cdot \cos(\alpha)$ $z' := z$
Rotation um x-Achse	$x' := x$ $y' := y \cdot \cos(\alpha) - z \cdot \sin(\alpha)$ $z' := y \cdot \sin(\alpha) + z \cdot \cos(\alpha)$
Rotation um y-Achse	$x' := z \cdot \sin(\alpha) + x \cdot \cos(\alpha)$ $y' := y$ $z' := z \cdot \cos(\alpha) - x \cdot \sin(\alpha)$
Rotation um beliebige Achse	Überführe durch Translation und Rotation die Drehachse auf eine der drei Hauptachsen; rotiere; transformiere zurück.

3D-Transformationen lassen sich beschreiben als 4×4-Matrizen, mit denen die homogenen Koordinaten eines Punktes multipliziert werden.

Die homogenen Koordinaten eines Punktes $P = (x, y, z)$ lauten $[x \cdot w, y \cdot w, z \cdot w, w]$ mit $w \neq 0$ (z.B. $w = 1$). Die homogenen Koordinaten eines Richtungsvektors $R = (x, y, z)$ lauten $[x, y, w, 0]$.

Die Transformationsmatrizen für die Translation, Skalierung und Rotation lauten:

Translation:

$$[x', y', z', 1] := [x, y, z, 1] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

Skalierung:

$$[x', y', z', 1] := [x, y, z, 1] \cdot \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um z-Achse:

$$[x', y', z', 1] := [x, y, z, 1] \cdot \begin{bmatrix} \cos(\alpha) & \sin(\alpha) & 0 & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um x-Achse:

$$[x', y', z', 1] := [x, y, z, 1] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) & 0 \\ 0 & -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um y-Achse

$$[x', y', z', 1] := [x, y, z, 1] \cdot \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

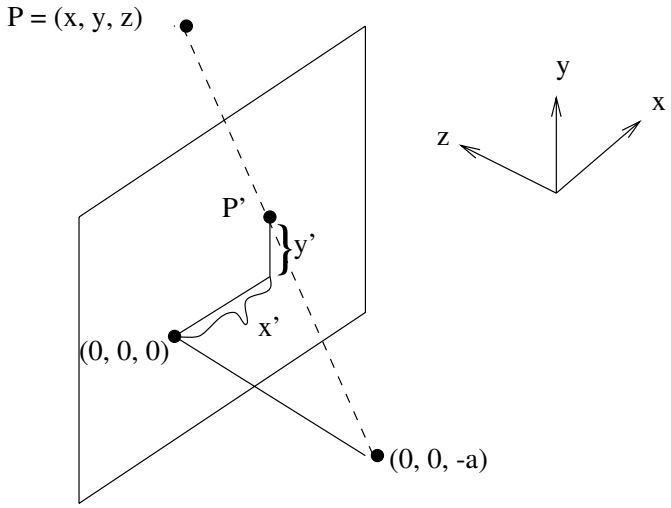
Hintereinander auszuführende Transformationen lassen sich durch das Produkt der beteiligten Transformationsmatrizen beschreiben.

7.4 Projektion

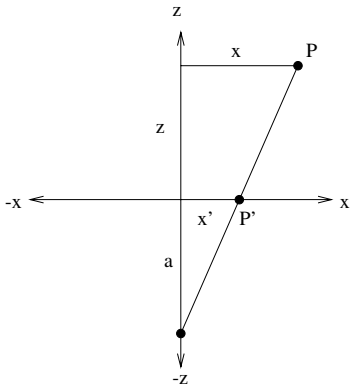
Gegeben sind

- das zu projizierende Objekt
- die Bildebene
- das Projektionszentrum (Augenpunkt).

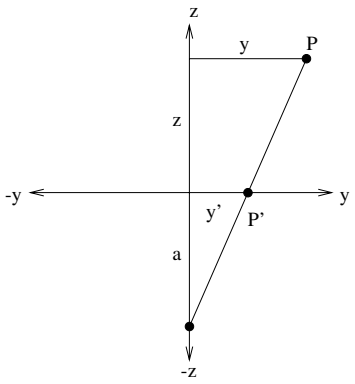
Der Punkt habe die Koordinaten (x, y, z), die Bildebene sei die $x - -y$ - Ebene, der Augenpunkt sei bei $(0, 0, -a)$.



Gesucht sind die Koordinaten x', y' von Punkt P'.



Blick "von oben" auf die $x - z$-Ebene.

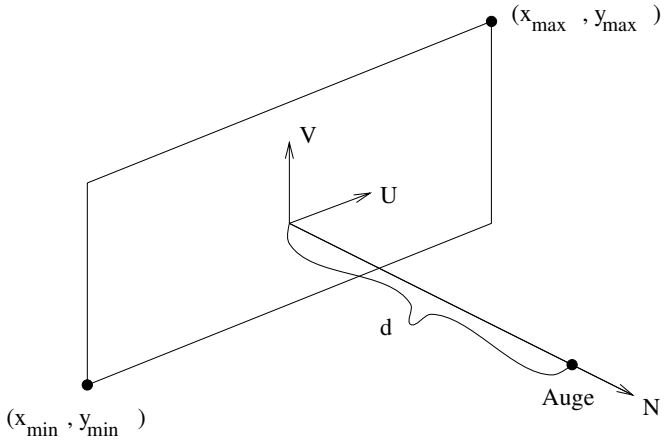


Blick "von der Seite" auf die $y - z$-Ebene.

Aufgrund der Strahlensätze ergibt sich

$$x' := \frac{x}{1 + z/a}, \; y' := \frac{y}{1 + z/a}$$

Synthetische Kamera
Die synthetische Kamera wird beschrieben durch drei orthogonale Vektoren U, V, N, eine Entfernung d sowie Eckpunkte eines Fensters $x_{\min}, y_{\min}, x_{\max}, y_{\max}$.



Die Vektoren U und V legen die Bildebene fest, Vektor N bestimmt die Blickrichtung. Über die Eckpunkte des Fensters wird der sichtbare Teil definiert (entspricht der Brennweite eines Kameraobjektivs). Der sichtbare Teil kann noch weiter eingeschränkt werden durch Angabe einer `front plane` und `back plane`.

Viewing Pipeline
Jedes Objekt sei beschrieben durch Modellkoordinaten. Seine Orientierung in der Szene wird beschrieben durch eine Transformationsmatrix mit akkumulierten Schiebe-, Dreh- und Skalieranteilen.

Jedes Polygon durchläuft folgende Pipeline:

MC → WC beschreibe Szene im *World Coordinate System* durch Anwendung der objektbezogenen Transformationsmatrizen auf alle Polygonpunkte.

WC → VRC beschreibe Szene im *View Reference Coordinate System* (d.h. aus der Sicht der synthetischen Kamera) durch Wechsel eines Koordinatensystems.

VRC → VRC bestimme den sichtbaren Teil durch *Polygonclipping* an den 6 Flächen des durch die synthetische Kamera definierten Pyramidenstumpfes.

VRC → PC beschreibe Szene im *Projection Coordinate System* durch perspektische Projektion (z-Werte behalten).

PC → DC beschreibe Szene im *Device Coordinate System* durch Abbildung der Projektionskoordinaten auf Bildschirmpunkte.

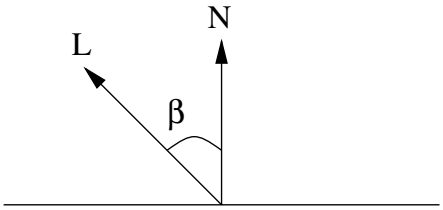
Zur Anzeige als Drahtgittermodell können nun die projizierten Polygonpunkte durch Geraden verbunden werden.
Für eine fotorealistische Darstellung benötigt der Renderer Angaben über Materialbeschaffenheit und Normalenvektor (in Weltkoordinaten) jeder Polygonfläche sowie Art und Position der für die Szene verwendeten Lichtquellen. Vor dem zeilenweisen Füllen der projizierten Objektpolygone werden diese triangulisiert, so daß sich alle weiteren Operationen auf 2-dimensionale Dreiecke beziehen.

7.5 Rendering

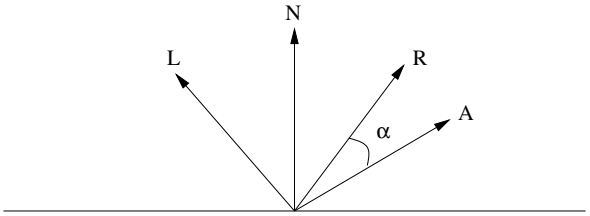
In die Berechnung des Farbwertes eines Pixels gehen die vorhandenen Lichtquellen, der Augenpunkt, die Oberflächennormale und die Materialeigenschaften ein. Die Materialeigenschaften sind gegeben durch

- die Farbe
- Hintergrundbeleuchtung (unabhängig von Lichtquellen)
- Spiegelungskraft (regelt diffuse versus spekulare Reflexion)
- Rauheit (regelt Größe der Glanzlichter)
- Transparenz (regelt Ausmaß der Lichtdurchlässigkeit)
- Lichtbrechung (regelt Verzerrung bei transparenten Objekten)

In den diffusen Anteil geht die Intensität der Lichtquelle und der Kosinus des Winkels β zwischen Oberflächennormale N und Vektor in Richtung Lichtquelle L ein:



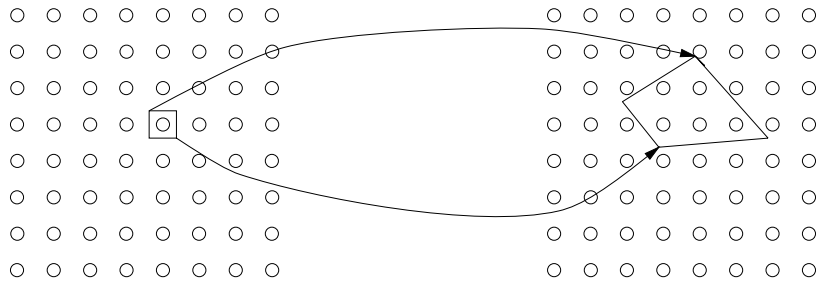
In den spekularen Anteil geht die Intensität der Lichtquelle ein, der Kosinus des Winkel α zwischen dem reflektierten Strahl R und dem Vektor A in Richtung Augenpunkt und die Rauheit.



Texture Mapping
Zur realistischen Gestaltung von Oberflächen verwendet man ein zweidimensionales Musterfeld (*texture map*), aus dem für jedes Pixel die zugehörige Farbe ermittelt werden kann. Zugrunde gelegt sind zwei Abbildungen

$$\begin{array}{lll} \text{Musterraum} & (u, v) & \\ \downarrow & & \text{Abbildung des Musters} \\ \text{Objektraum} & (x_0, y_0, z_0) & \\ \downarrow & & \text{Projektion} \\ \text{Bildraum} & (x, y) & \end{array}$$

Die Verknüpfung der zugehörigen inversen Transformationen liefert die zum Einfärben eines Pixels benötigte Information.



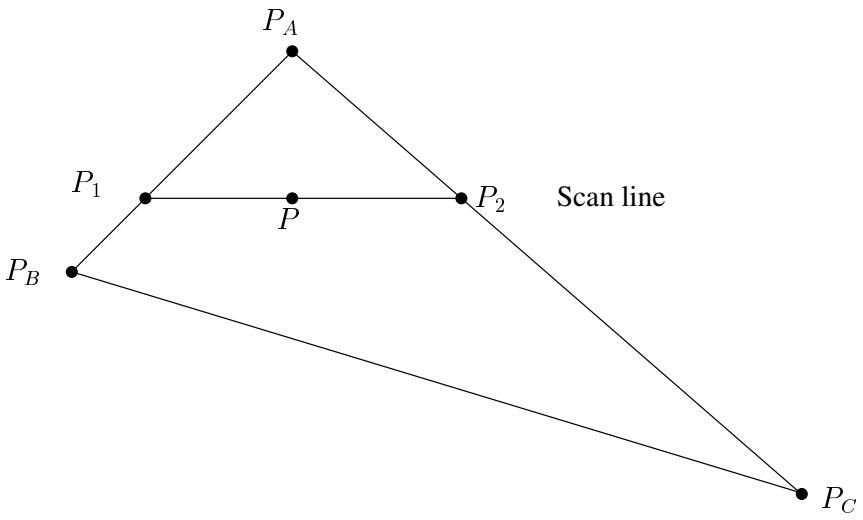
Zum Einfärben im Bildraum wird die gewichtete Durchschnittsintensität aller überdeckten Pixel im Musterraum benutzt.

Bump Mapping
Um den Eindruck von unebenen Oberflächen zu erwecken, werden in einem Graubild Informationen zur Veränderung des Normalenvektors abgelegt.

Die Schattierungsalgorithmen unterscheiden sich bzgl. des Aufwandes zur Bestimmung der Helligkeitswerte im Innern eines 2D-Dreiecks.

Flat Shading
Alle Punkte des Dreiecks erhalten dieselbe Farbe. Berücksichtigt werden Hintergrundbeleuchtung und diffuse Reflexion.

Gouraud Shading
Es seien pro Dreieck an den Eckpunkten die Normalen in Weltkoordinaten bekannt (d.h. bei einer gekrümmten Fläche sind die drei Normalenvektoren verschieden).
Für jedes Pixel im Innern des Dreiecks wird nun durch doppelte Interpolation die Intensität approximiert:

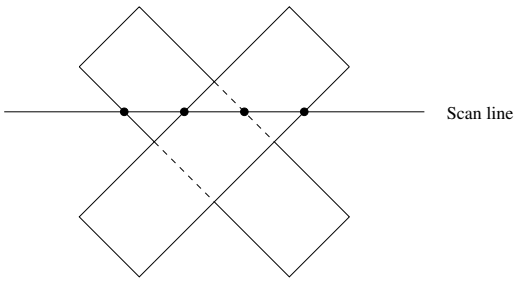


Phong Shading
Es seien pro Dreieck an den Eckpunkten die Normalen in Weltkoordinaten bekannt. Für jedes Pixel im Innern des Dreiecks wir nun durch doppelte Interpolation seine individuelle Normale approximiert und dann der Farbwert berechnet.

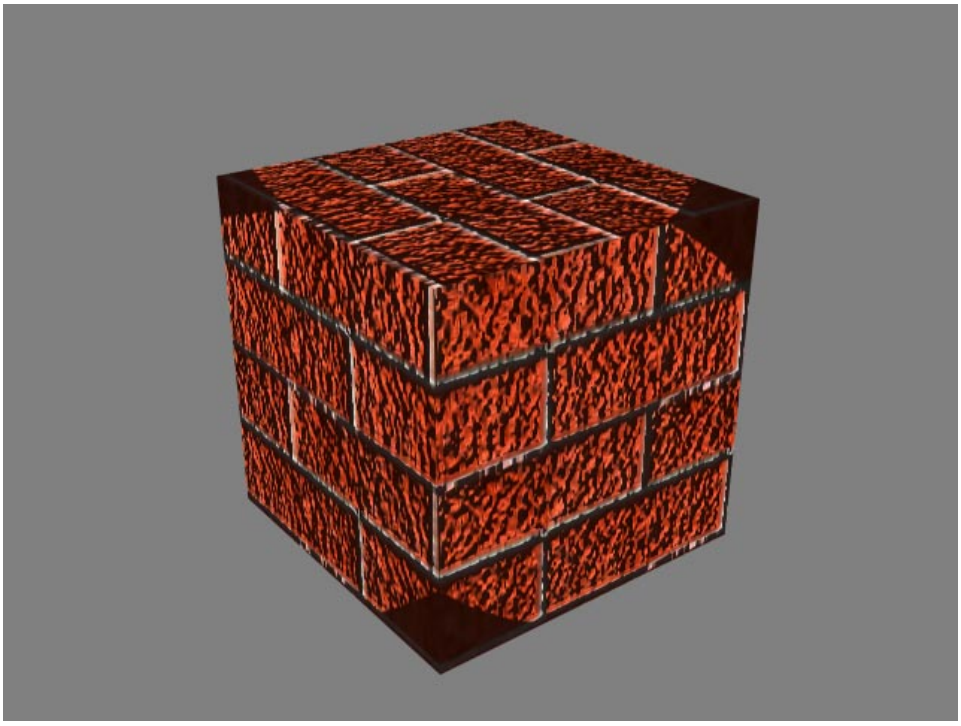
Sichtbarkeit
Nicht sichtbare Flächen oder Flächenanteile werden wie folgt entfernt:
Beim *Back Face Culling* wird aufgrund des Normalenvektors einer Fläche entschieden, ob sie vom Betrachter abgewandt ist und somit vom Rendern ausgeschlossen werden kann.
Beim z-Buffer-Algorithmus wird zusätzlich zum Bildwiederholspeicher `bild` ein 2-dimensionales Feld `tiefe` gehalten, welches für jedes Pixel den z-Wert des in diesem Punkt dem Betrachter am nächsten liegenden Objekts enthält.

```
Fuer jede Flaeche tue:
Fuer jedes Pixel (x, y) auf dieser Flaeche tue:
   berechne Farbe x und Tiefe z
   falls z < tiefe[x, y] dann
      bild  [x, y] := c
      tiefe [x, y] := z
```

Beim Spanning-Scan-line-Algorithmus durchläuft eine Scan-Line von oben nach unten das Bild und stützt sich auf eine nach y-Koordinaten sortierten Liste von Polygonen. Alle Kanten der aktiven Polygone befinden sich in einer nach x-Koordinaten sortierten Liste. Jeweils am Schnittpunkt der Scan-Line mit einer Polygonkante wird über die Sichtbarkeit der nächsten Polygonfläche entschieden.



Schatten
Zur Berechnung von Schatten eignen sich alle *Hidden-Surface*-Algorithmen, da die verdeckten Flächen einer Szene genau den beschatteten Flächen entsprechen, wenn die Position der Lichtquelle und des Betrachters zusammenfallen.
Zunächst wird daher als Phase 1 aus der Sicht der Lichtquelle die Szene in einen Schattentiefenpuffer abgebildet. Phase 2 berechnet dann für den jeweiligen Betrachtungsstandpunkt die Szene mit einem modifizierten Tiefenpuffer-Algorithmus: Ergibt die Überprüfung des z-Wertes mit dem Eintrag `tiefe[x, y]` im Tiefenpuffer, daß dieses Pixel sichtbar ist, so wird der Punkt $P(x, y, z)$ in den Koordinatenraum von Phase 1 transformiert. Ist die z-Koordinate z' des transformierten Punktes P kleiner oder gleich dem Eintrag $[x', y']$ im Schattentiefenpuffer, dann liegt der Punkt P nicht im Schatten, andernfalls liegt er im Schatten, und seine Intensität muß entsprechend reduziert werden.

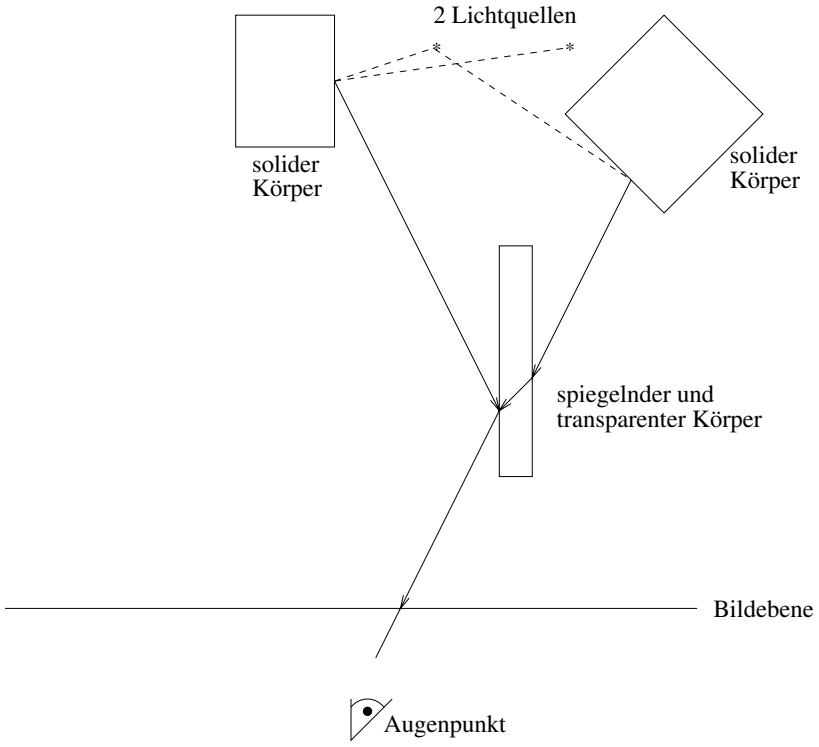


Würfel, gerendert mit Texturemapping, Bumpmapping und Spotlicht

Kugel, gerendert mit spiegelnder Oberfläche und Environment-Textur

7.6 Ray Tracing

Im Gegensatz zum Objektraum-bezogenen Rendern können beim Bildraum-bezogenen Ray Tracing Lichtspiegelungen und Lichtbrechungen berücksichtigt werden. Außerdem lassen sich mathematisch beschreibbare Körper (z.B. Kugel) ohne den Umweg über Approximationspolygone präziser darstellen. Ausgehend vom Augenpunkt wird durch jedes Pixel ein Strahl gelegt und der Schnittpunkt dieses Strahls mit dem ersten getroffenen Objekt bestimmt. Trifft der Strahl auf kein Objekt, so erhält das Pixel die Hintergrundfarbe. Ist das Objekt spiegelnd, so wird der Reflexionsstrahl berechnet und rekursiv weiterbehandelt. Ist das Objekt transparent, wird zusätzlich der gebrochene Strahl weiterbehandelt. Zur Berechnung von Schatten wird von jedem Schnittpunkt zwischen Strahl und Objekt zu jeder Lichtquelle ein zusätzlicher Strahl ausgesandt. Trifft dieser Strahl ein blockierendes Objekt, dann liegt der Schnittpunkt im Schatten dieser Lichtquelle, und das von ihr ausgestrahlte Licht geht in die Intensitätsberechnung des Punktes nicht ein.



Prinzip der Strahlverfolgung

7.7 Caligary trueSpace

Caligary trueSpace ist ein Werkzeug zum Modellieren, Rendern und Animieren von 3D-Objekten.
Die charakteristischen Leistungsmerkmale lauten:

3D-Editor unter Verwendung von

- hierarchischer Objektstruktur,
- Polygonvereinigung, -schnitt, -komplement,
- Translation, Skalierung, Rotation,
- Extrusion,
- Verformung,
- Drehkörpern,
- Kurven.

Projektion und Rendern unter Berücksichtigung von

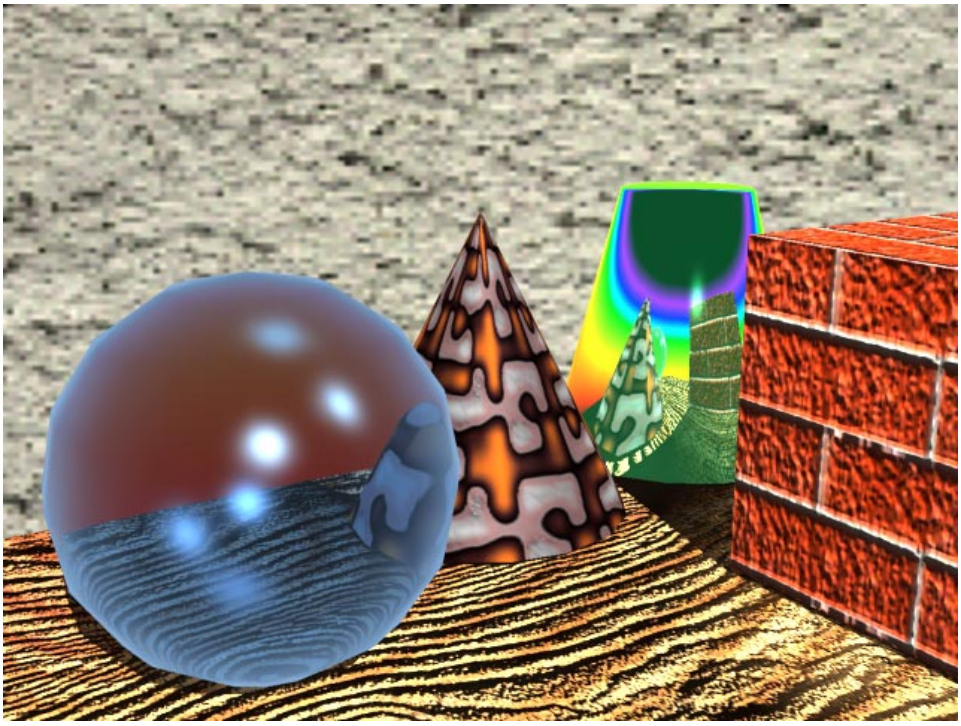
- Lichtquellen,
- Kamerastandpunkten,
- Spiegelung, Transparenz, Lichtbrechung,
- Texture Mapping, Bump Mapping.

Animation durch

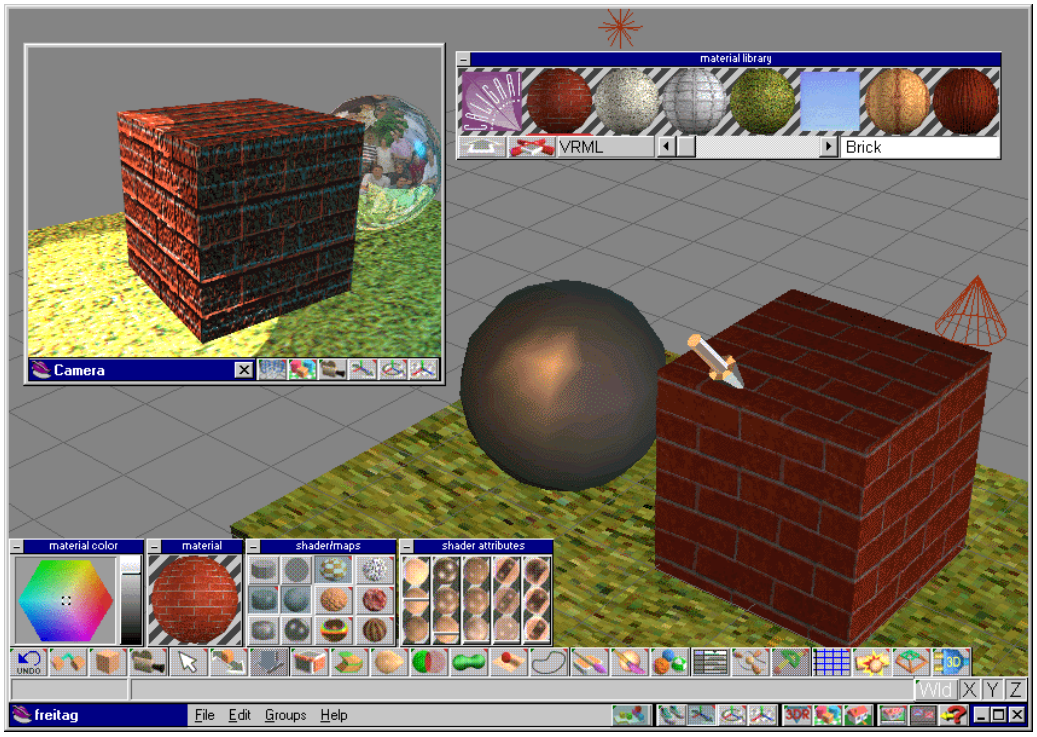
- Keyframes,
- Pfade.

Ausgabe als

- Bitmapdateien,
- VRML-Welten,
- AVI-Files.



Szene mit 5 Objekten, berechnet mit Raytracer unter Verwendung von Reflektion, Transparenz, Texture-Mapping, Bump-Mapping, Hintergrund-Mapping



Screenshot vom 3D-Werkzeug Caligary trueSpace

Kapitel 8

VRML

VRML, sprich Wörmel, ist eine für das WWW entworfene Virtual Reality Modelling Language zur Beschreibung von 3-dimensionalen Szenen mit multimedialen Komponenten und Animation. Die gerenderte Projektion der Szene kann von jedem Web-Browser betrachtet werden, der über ein passendes Plugin verfügt.

8.1 Geschichte

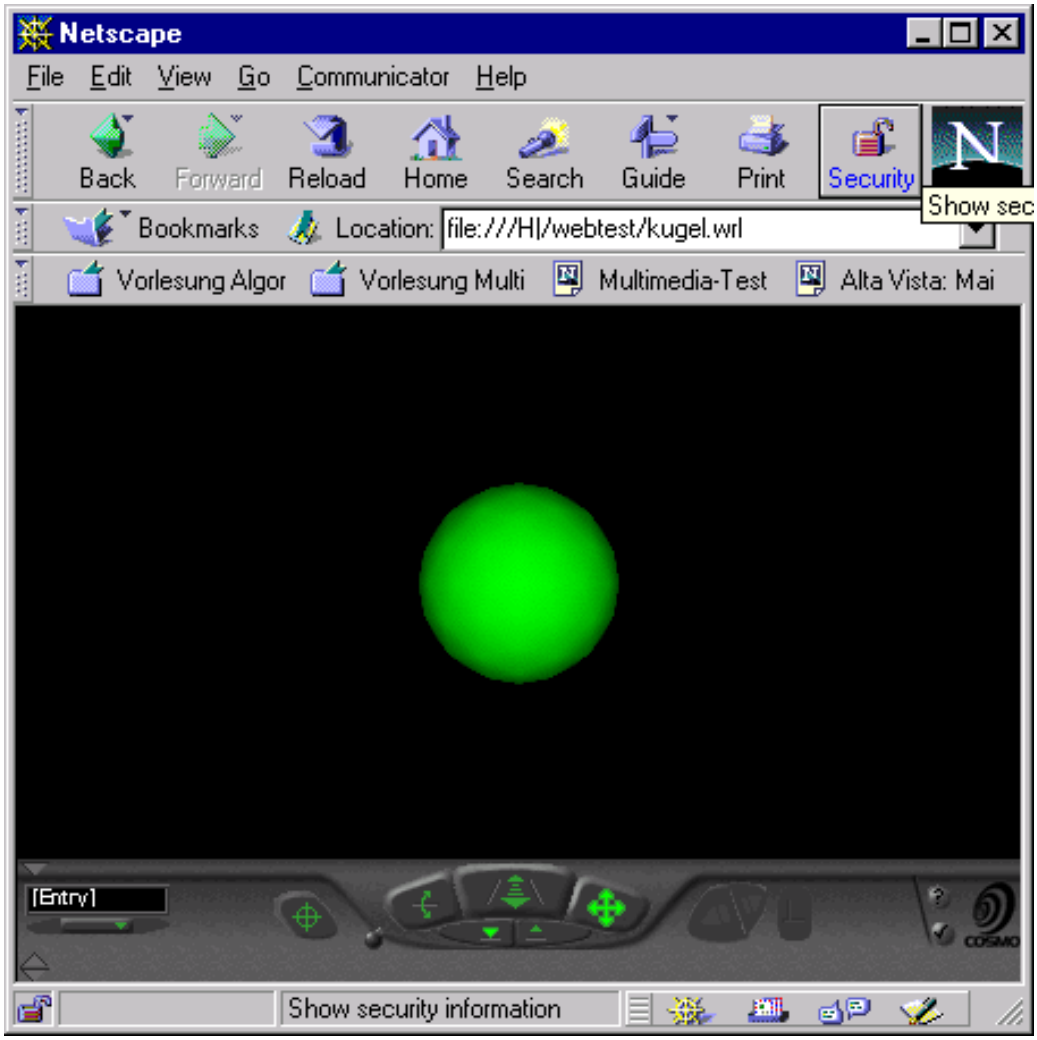
Bei der Nutzung von Computer-Resourcen läßt sich ein weiterer Paradigmenwechsel beobachten: Während in der EDV-Gründerzeit speziell ausgebildete Techniker am Mainframe-Computer Befehle eintippten, danach einige Jahrzehnte später Kopfarbeiter per Drag & Drop Fensterelemente manipulierten, surft inzwischen jedermann und -frau in einer weltweit vernetzten multimedialen Landschaft. Der Kontext hat sich also von institutionell über persönlich zu sozial gewandelt.
Aus diesem Zeitgeist heraus diskutierten im April 1994 auf der 1st International WWW Conference in Genf Tim Berners-Lee, einer der Väter des WWW, mit den beiden Autoren des Systems Labyrinth, Mark Pesce und Tony Parisi. Es wurde eine Mailing List aufgesetzt, die schon nach wenigen Wochen mit mehr als 1000 Teilnehmern über Syntax für Strukturen, Verhalten und Kommunikation debattierte. Bereits im Oktober 1994 wurde auf der 2nd International WWW Conference in Chicago VRML 1.0 vorgestellt, ein Entwurf, der wesentlich vom Silicon Graphics System Open Inventor inspiriert war. VRML 1.0 konnte bereits geometrische Grundkörper und Polygone in einem Koordiantensystem plazieren und ihre Farbe und Materialeigenschaften spezifizieren. Auch ließen sich durch Hyperlinks Objekte beliebig im Web referieren. Abgesehen von dieser Möglichkeit der Interaktion handelte es sich allerdings um rein statische Szenen.
Diesem Manko sollte eine schnellstens eingerichtete VAG (VRML Architecture Group) abhelfen, welche Überlegungen zur Animation und zur Integration multimedialer Komponenten wie Sound und Video koordinierte. Zur 1st International VRML Conference im Dez. 1995 war es dann soweit: Als Sieger einer Ausschreibung für VRML 97 ging Moving Worlds von Silicon Graphics nach einer On-Line-Wahl. Überarbeitete Syntax sorgte für die Beschreibung statischer Szenen mit multimedialen Bestandteilen und ein neues Event-Handling-Konzept erlaubte Animation dieser Szenen sowie Interaktion mit dem Benutzer.

8.2 Einbettung

VRML-Szenen werden beschrieben in ASCII-Dateien mit der Endung *.wrl, welche innerhalb einer HTML-Seite mit dem EMBED-Kommando referiert werden, z.B.

```
<EMBED SRC="kugel.wrl "WIDTH=400 HEIGTH=300>
```

Ein entsprechend konfigurierter Web-Server schickt dem anfordernden Clienten als Vorspann dieser Daten den Mime-Typ VRML, worauf das zur Betrachtung installierte Plugin, z.B. Cosmo Player 2.0 von Silicon Graphics, die eingehenden Daten in eine interne Datenstruktur einliest, von wo sie zur Berechnung einer fotorealistischen Projektion verwendet werden. In welcher Weise Blickwinkel und Orientierung in der Szene modifiziert werden können, bleibt dem Plugin überlassen: Mit Mauszeiger und Keyboard Shortcuts wandert der Benutzer durch eine virtuelle Welt, verkörpert im wahrsten Sinne des Wortes durch einen Avatar, seiner geometrischen Repräsentation, beschränkt in seiner Beweglichkeit durch physikalische Restriktionen und durch eine simulierte Schwerkraft.



Screenshot vom VRML-Plugin

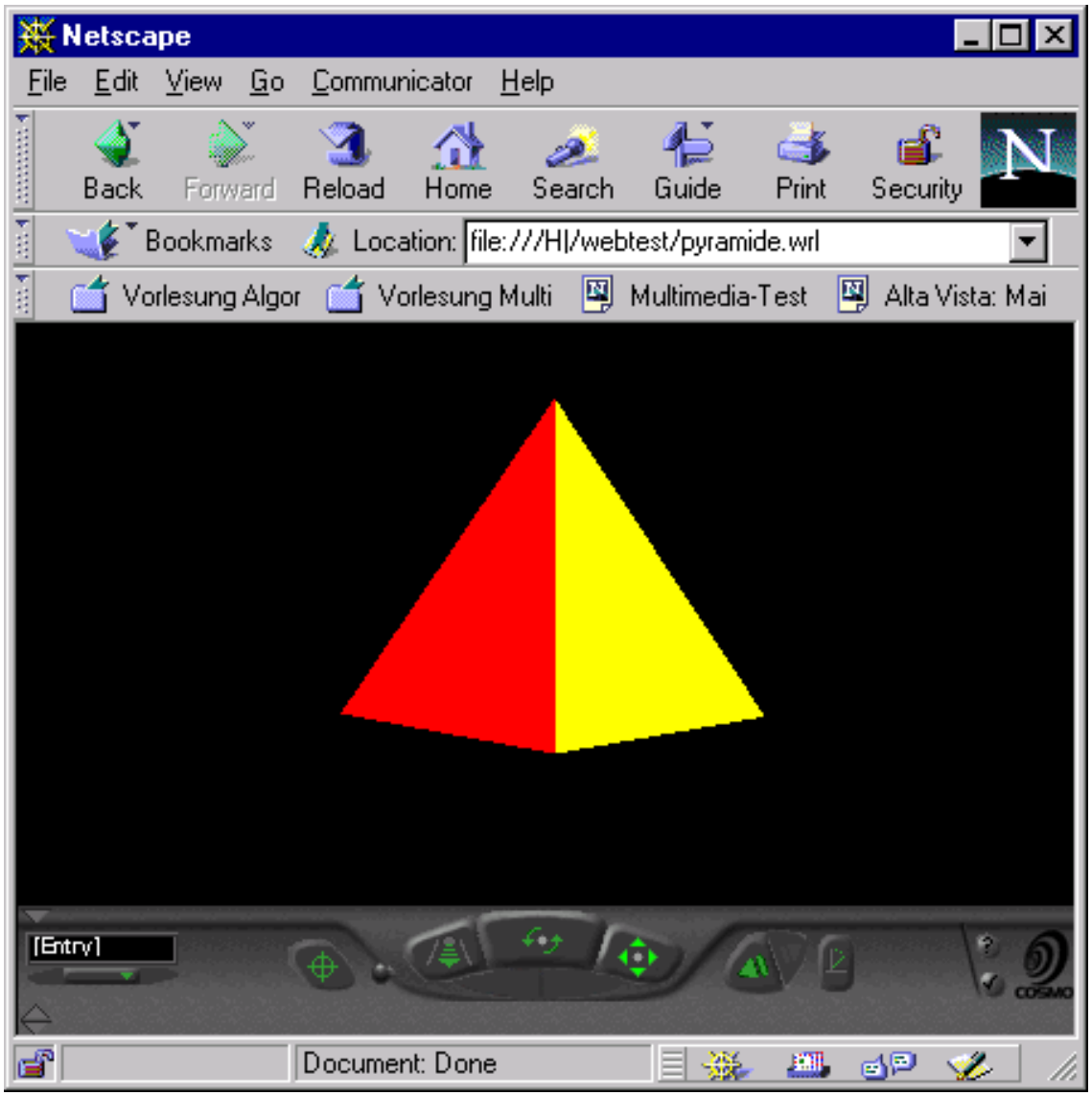
8.3 Geometrie

Wichtigster Bestandteil von VRML-Szenen ist der sogenannte Knoten (meistens mit großem Anfangsbuchstaben geschrieben) der ähnlich eines Programmiersprachenrecords aus Feldern verschiedenen Typs besteht (meistens klein geschrieben). Diese Felder verweisen entweder auf nicht weiter strukturierte Objektknoten oder andere Gruppenknoten, die wiederum mittels ihrer Felder verzweigen können.
Beispiel 1 zeigt den Aufbau einer Szene, in der eine Kugel mit Radius 1.5 im Ursprung des Weltkoordinatensystems plaziert wird. Die x-Richtung entspricht der horizontalen Bewegung, y beschreibt die vertikale Richtung und z wächst auf den Betrachter zu. Der Sphere-Knoten hat dabei als einziges (optionales) Feld den Radius. Diese Kugel wird referiert über das geometry-Feld des Shape-Knotens, zuständig für die Gestaltung eines Objekts. über das appearance-Feld wird die Materialbeschaffenheit in Form einer RGB-Farbe und eines Transparenz-Koeffizienten spezifiziert. Der Shape-Knoten wiederum ist als eins der Kinder im Transform-Knoten eingetragen, der über ein translation-Feld für die Verschiebung der Kugel sorgt.

```
#VRML V2.0 utf8
# kugel.wrl
# gruene, stark reflektierende  Kugel mit Radius 1.5

Transform {                          # Plaziere
  translation 0 0 0                  # im Ursprung
  children [
    Shape {                          # eine Gestalt
      geometry Sphere {              # von der Form einer Kugel
        radius 1.5                   # mit Durchmesser 1.5
      }
      appearance Appearance {        # in der Erscheinung
        material Material {          # mit Materialbeschaffenheit
          diffuseColor 0 1 0         # gruene Farbe
          shininess 0.9              # stark reflektierend
        }
      }
    }
  ]
}
}
```

Neben den Grundbausteinen Sphere (Kugel), Box (Quader), Cone (Kegel) und Cylinder (Zylinder) lassen sich eigene geometrische Gebilde konstruieren. Ausgehend von einer Liste von 3-D-Punkten im Raum werden jeweils gegen den Uhrzeigersinn alle Punkte durchlaufen, die ein Face (durch Polygon approximierte Körperfläche) aufspannen. Beispiel 2 zeigt die Definition einer 5-farbigen Pyramide mit quadratischer Grundfläche.

```
#VRML V2.0 utf8
# pyramide.wrl
# selbstdefinierte 5-seitige Pyramide

Shape {
  geometry IndexedFaceSet {

    coord Coordinate {
      point [            # beteiligte Punkte
         0  3  0         # 0. Pyramidenpunkt (Spitze)
         0  0 -2         # 1. Pyramidenpunkt (Norden)
        -2  0  0         # 2. Pyramidenpunkt (Westen)
         0  0  2         # 3. Pyramidenpunkt (Sueden)
         2  0  0         # 4. Pyramidenpunkt (Osten )
      ]
    }

    coordIndex [         # Polygone gegen Uhrzeiger, Ende: -1
         4  3  2  1 -1   # 0. Face: Punkte 1 2 3 4 (Grundflaeche)
         0  1  2    -1   # 1. Face: Punkte 0 1 2   (Nordwesten)
         0  2  3    -1   # 2. Face: Punkte 0 2 3   (Suedwesten)
         0  3  4    -1   # 3. Face: Punkte 0 3 4   (Suedosten)
         0  4  1         # 4. Face: Punkte 0 4 1   (Nordosten)
    ]

    colorPerVertex FALSE
    color Color {
        color [          # pro Face eine Farbe benennen
          0 1 1          # 0. Face: Cyan
          1 0 0          # 1. Face: Rot
          1 1 0          # 2. Face: Gelb
          0 1 0          # 3. Face: Gruen
          0 0 1          # 4. Face: Blau
        ]
    }
  }
}
```

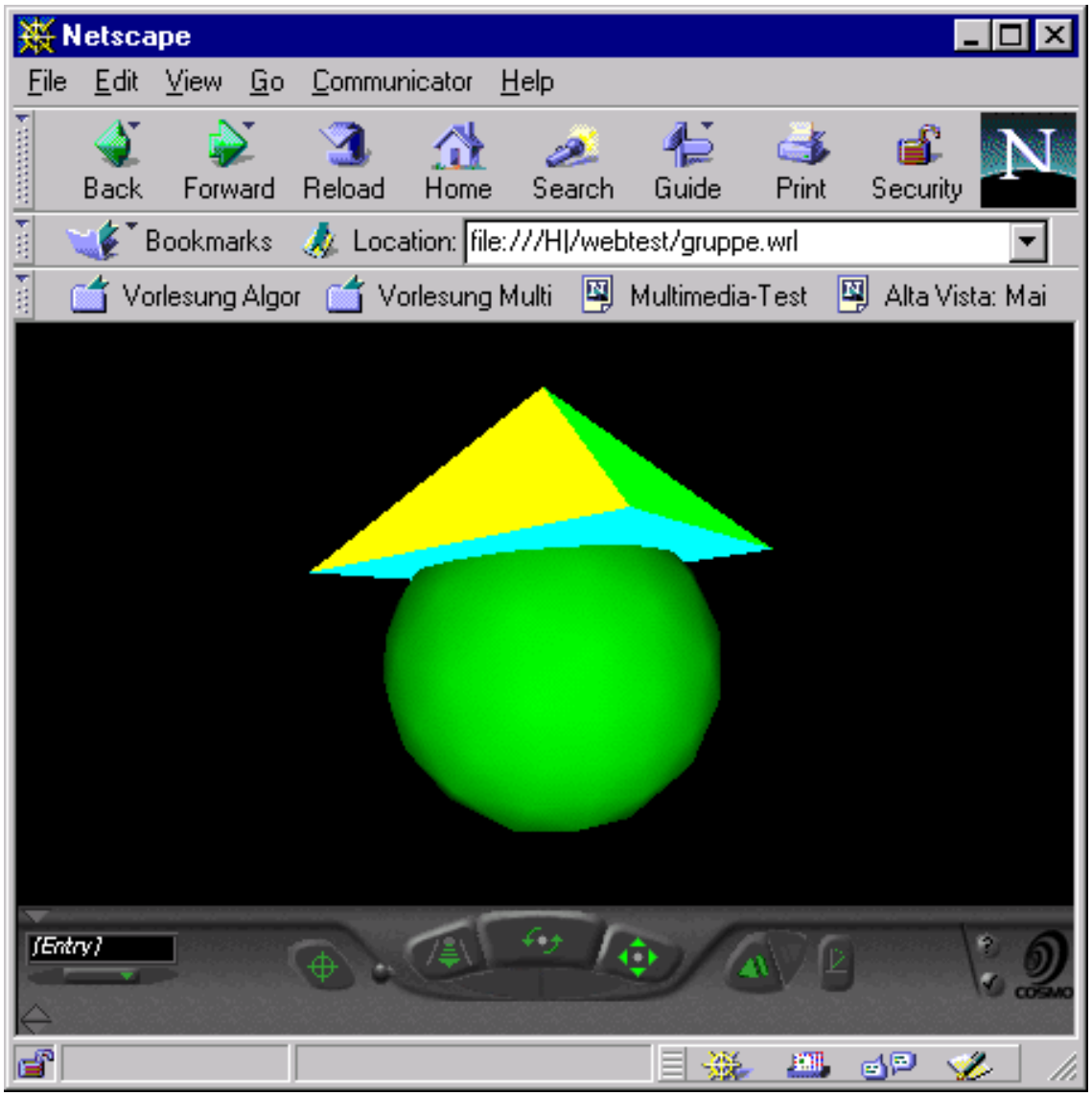


Screenshot vom VRML-Plugin

8.4 Wiederverwendung

Zur Reduktion der Dateigrößen und zur besseren Lesbarkeit lassen sich einmal spezifizierte Welten wiederverwenden. Beispiel 3 zeigt die Kombination der beiden graphischen Objekte Kugel und Pyramide, wobei die Pyramide leicht nach hinten gekippt oberhalb der Kugel positioniert wird. Ferner wurde ein Hyperlink eingerichtet, der zu einer weiteren VRML-Welt führt, sobald der Mauszeiger auf der Kugel gedrückt wird.

```
#VRML V2.0 utf8
# gruppe.wrl: Kugel mit Hyperlink unter gekippter Pyramide

Transform{                                  # Plaziere
  children[                                 # die folgenden Objekte:
    Anchor {                                # Hyperlink
      url "multimedia.wrl"                  # zur VRML-Datei textur.wrl
      description "Next world"              # Anzeige im Statusfenster
      children[                             # zugeordnet ist die Welt
        Inline {url "kugel.wrl"}            # aus Datei kugel.wrl
      ]
    }

    Transform {
      translation  0 1   0                  # hebe 1 Einheit nach oben
      scale        1 0.5 1                  # skaliere bzgl. y auf 50 %
      rotation     1 0   0 -0.523333        # kippe um 30 Grad nach hinten
      children[                             # verwende Welt
        Inline {url "pyramide.wrl"}         # aus Datei pyramide.wrl
      ]
    }

  ]
}
```

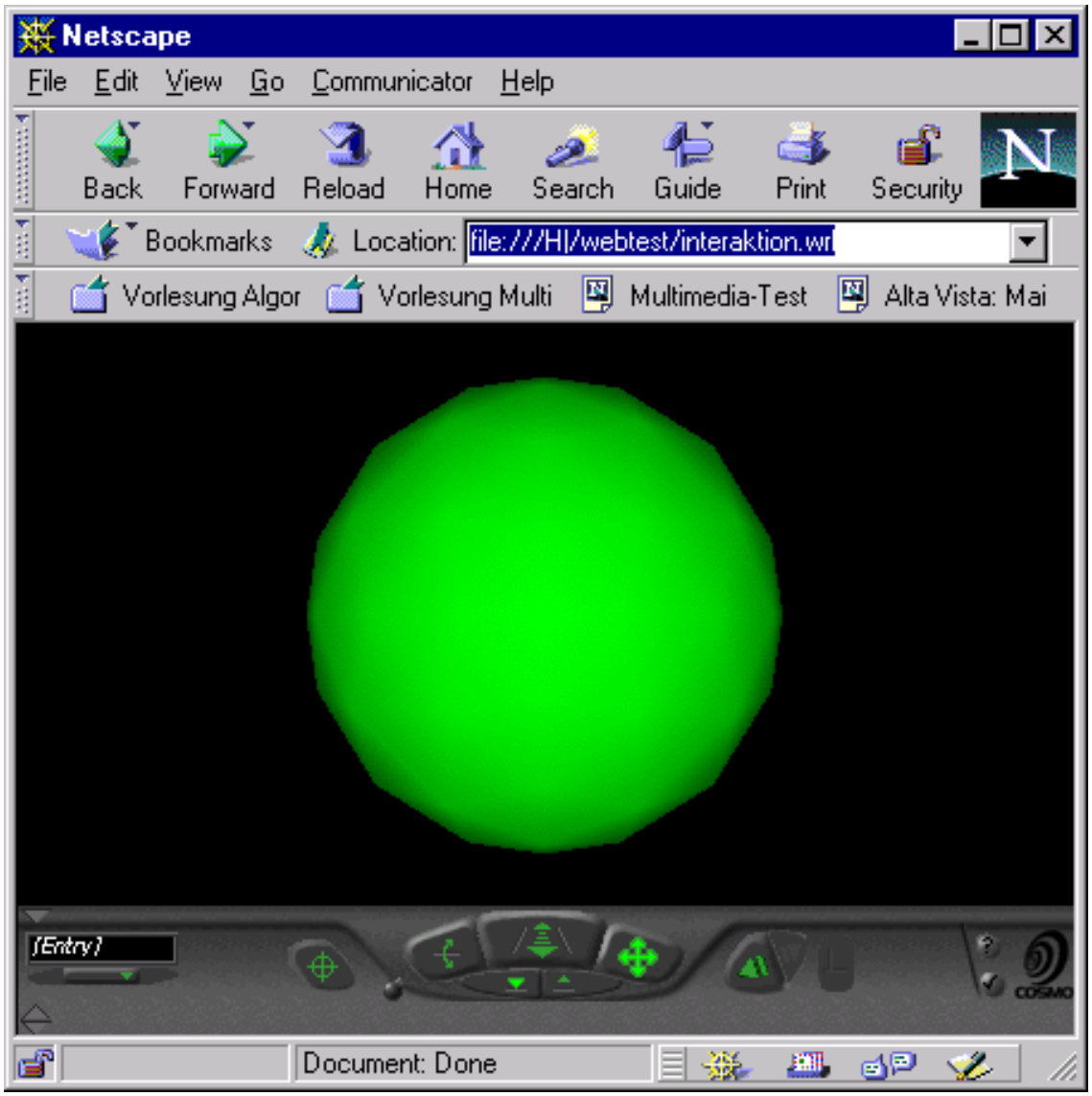


Screenshot vom VRML-Plugin

8.5 Interaktion

VRML97 bietet zahlreiche Möglichkeiten, mit denen einer Szene Dynamik und Interaktion verliehen werden kann. Die zentralen Bausteine für die hierzu erforderliche Ereignisbehandlung sind die EventIn- bzw. EventOut-Felder von Knoten, mit denen Meldungen empfangen und Zustandsänderungen weitergeschickt werden können. Es gibt Time,- Proximity-, Visibility-, Collision- und Touch-Sensoren, welche das Verstreichen einer Zeitspanne, das Annähern des Benutzers, die Sichtbarkeit von Objekten, das Zusammentreffen des Avatars mit einem Objekt und die Berührung mit dem Mauszeiger signalisieren. Verständlicherweise müssen Typ des verschickenden Ereignisfeldes und Typ des empfangenden Ereignisfeldes übereinstimmen. Beispiel 5 zeigt die Kugel versehen mit einem Touch-Sensor, welcher bei Mausdruck eine Nachricht an den Soundknoten schickt, der auf diese Weise seinen Spielbeginnzeitpunkt erhält und die zugeordnete Wave-Datei startet.

```
#VRML V2.0 utf8
# interaktion.wrl:
# Kugel macht Geraeusch bei Beruehrung

Group {                                 # plaziere Gruppenknoten
  children [                            # bestehend aus
    DEF Taste TouchSensor {}            # einem Touch-Sensor
    Inline { url "kugel.wrl" }          # und einer Kugel
  ]
}

Sound {                                 # plaziere Soundknoten
  source DEF Tut AudioClip {            # gespeist von Audio-Clip
    url "tut.wav"                       # aus der Wave-Datrei tut.wav
  }
  minFront  5                           # Anfang des Schallbereichs
  maxFront 50                           # Ende des Schallbereichs
}

ROUTE Taste.touchTime                   # bei Beruehrung der Kugel
   TO Tut.set_startTime                 # schicke Systemzeit an den Knoten Tut
```

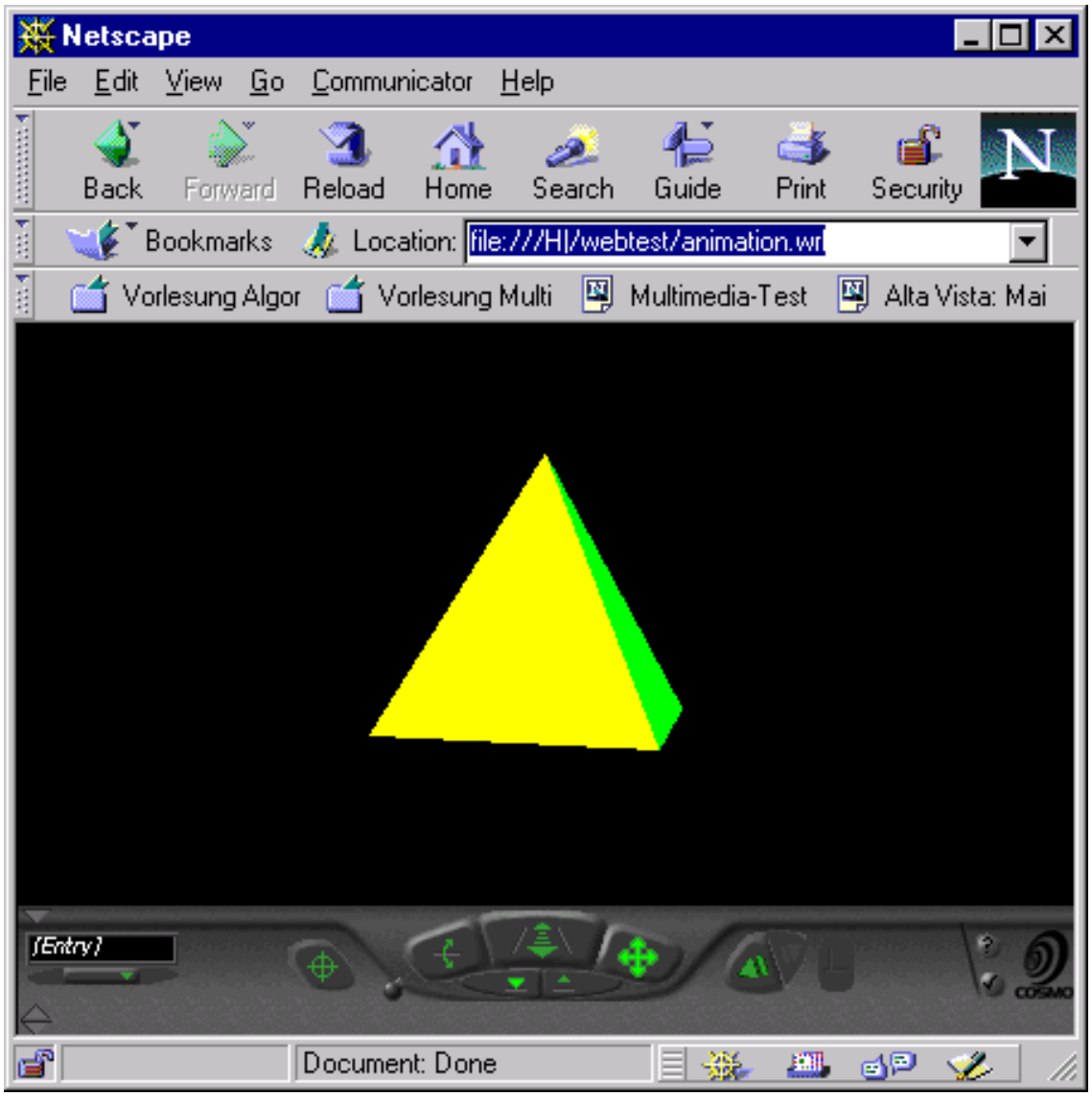


Screenshot vom VRML-Plugin

8.6 Animation

Die benutzergesteuerte oder automatische Bewegung von Objekten und Szenenteilen wird wiederum mit Hilfe der Ereignisbehandlung organisiert. Im Beispiel 6 wird die Ziehbewegung des gedrückten Mauszeigers zur Manipulation der lokalen Translation eines Objekts verwendet und das regelmäßige Verstreichen eines Zeitintervalls löst eine Nachricht an denselben geometrischen Knoten aus, der hierdurch seinen aktuellen Rotationsvektor erhält. Eine solche Konstruktion verlangt einen Orientation-Interpolator, welcher beliebige Bruchzahlen zwischen 0 und 1 auf die zugeordneten Werte seines Schlüsselintervalls abbildet, hier bestehend aus allen Drehwinkeln zwischen 0 und 3.14 (=180 Grad beschrieben in Bogenmaß), bezogen auf die y-Achse.

```
#VRML V2.0 utf8
# animation.wrl
# selbstaendig sich drehende und interaktiv verschiebbare Pyramide

DEF Schieber PlaneSensor {}            # Sensor zum Melden einer Mausbewegung

DEF Timer TimeSensor {                 # Sensor zum Melden eines Zeitintervalls
  cycleInterval 5                      # Dauer 5 Sekunden
  loop TRUE                            # Endlosschleife
}

DEF Rotierer OrientationInterpolator{  # Interpolator fuer Rotation
  key      [0 , 1]                     # bilde Schluessel 0 und 1 ab auf
  keyValue [ 0  1  0  0                #   0 Grad Drehung bzgl. y
             0  1  0  3.14]            # 180 Grad Drehung bzgl. y
}

DEF Pyramide Transform {               # plaziere Objekt mit Namen Pyramide
      children [                       # bestehend aus
        Inline {url "pyramide.wrl"}    # VRML-Datei pyramide.wrl
      ]
}

ROUTE Timer.fraction_changed           # falls Zeitintervall sich aendert
   TO Rotierer.set_fraction            # schicke Bruchteil an Rotierer
ROUTE Rotierer.value_changed           # falls Drehung sich aendert
   TO Pyramide.set_rotation            # schicke Drehwert an Pyramide

ROUTE Schieber.translation_changed     # falls gedrueckter Mauszeiger bewegt wird
  TO Pyramide.set_translation          # schicke Translationswert an Pyramide
```



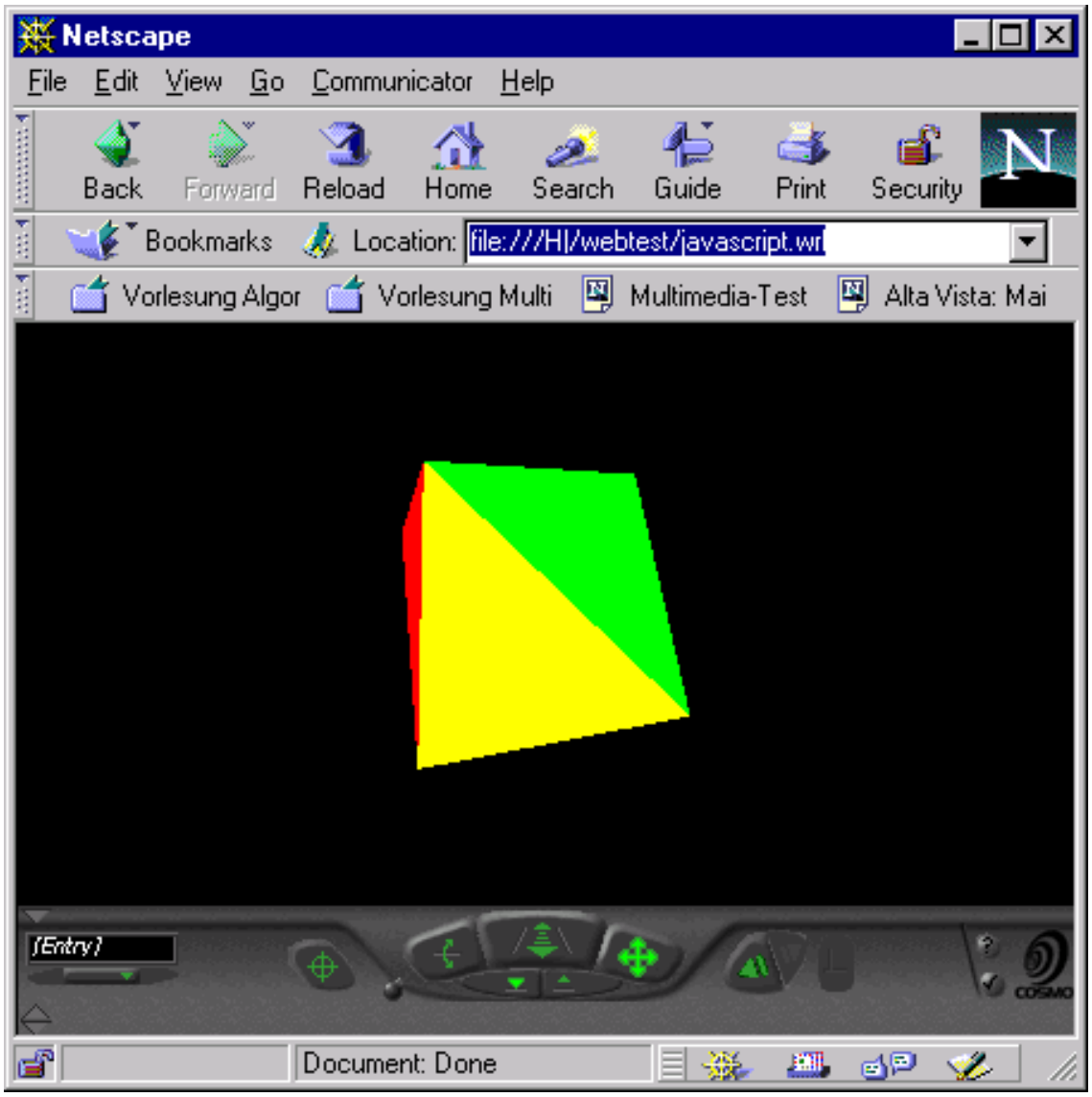
Screenshot vom VRML-Plugin

8.7 Scripts

Manchmal reichen die in VRML angebotenen Funktionen wie Sensoren und Interpolatoren nicht aus, um ein spezielles situationsbedingtes Interaktionsverhalten zu erzeugen. Abhilfe schafft hier der sogenannte Script-Knoten, welcher Input empfangen, Daten verarbeiten und Output verschicken kann. Z.B. kann eine vom Touch-Sensor geschickte Nachricht eine Berechnung anstoßen, deren Ergebnis in Form einer Translations-Nachricht an ein bestimmtes Objekt geschickt und dort zur Neupositionierung genutzt wird.
Die Formulierung des Berechnungsalgorithmus geschieht entweder durch ein Javascript-Programm, inline gelistet im Script-Knoten, oder durch eine assoziierte Java-Klasse, welche in übersetzter Form mit Dateiendung *.class lokal oder im Netz liegt. Zum Übersetzen der Java-Quelle ist das EAI (External Authoring Interface) erforderlich, welches in Form einiger Packages aus dem Verzeichnis importiert wird, in welches sie das VRML-Plugin bei der Installation deponiert hatte.
Beispiel 7 zeigt die Pyramide zusammen mit einem Java-Script, welches bei jedem Aufruf den Drehwinkel bzgl. der y-Achse um weitere 10 Grad erhöht.

```
#VRML V2.0 utf8
# javascript.wrl:
# Rotation eines Objekts wird ueber Javascript manipuliert

Group {                                  # gruppiere
     children [                          # folgende Objekte
       DEF Taste TouchSensor{}           # Beruehrungssensor Taste
       DEF Pyramide Transform {          # Objekt Pyramide
         children[                       # bestehend aus
           Inline {url "pyramide.wrl"}   # VRML-Welt pyramide.wrl
         ]
       }
     ]
   }

   DEF Aktion Script {                     # Script mit Namen Aktion
     eventIn  SFBool     isActive          # Input-Parameter
     eventOut SFRotation drehung           # Output-Parameter
     url [                                 # gespeist
        "javascript:                      //  von inline-Javascript
        function isActive(eventValue) {   // fuer den Event-In zustaendig
          if (eventValue == true) {       // falls isActive den Wert wahr hat
            drehung[0] += 0.3;            // drehe weiter bzgl. x-Achse
            drehung[1] += 0.2;            // drehe weiter bzgl. y-Achse
            drehung[2] += 0.1;            // drehe weiter bzgl. z-Achse
            drehung[3] += 0.174444;       // erhoehe Drehwinkel um 10 Grad
          }
        } "
     ]
   }
```

```
ROUTE Taste.isActive                # bei Beruehren der Pyramide
   TO Aktion.isActive               # sende Nachricht an das Script Aktion

ROUTE Aktion.drehung                # vom Script Aktion erzeugter Dreh-Vektor
   TO Pyramide.set_rotation         # wird an die Pyramide geschickt
```



Screenshot vom VRML-Plugin

8.8 Multiuser

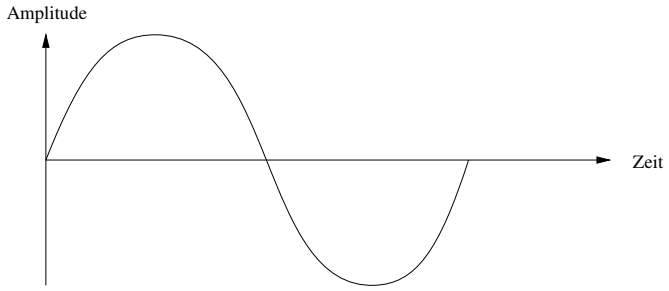
Eines der ursprünglichen Entwicklungsziele von VRML bleibt auch bei VRML 97 offen: es gibt noch keinen Standard für Multiuser-Welten. Hiermit sind Szenen gemeint, in denen mehrere Benutzer gleichzeitig herumwandern und interagieren können. Da jedes ausgelöste Ereignis von allen Beteiligten wahrgenommen werden soll, muß ein zentraler Server den jeweiligen Zustand der Welt verwalten und den anfragenden Klienten fortwährend Updates schicken. In diesem Zusammenhang erhält der Avatar eine aufgewertete Rolle: Zusätzlich zu seiner geometrischen Räumlichkeit, die schon zur Kollisionsdetektion in einer Single-User-Szenerie benötigt wurde, muß nun auch sein visuelles Äußeres spezifiziert werden, sicherlich ein weiterer wichtiger Schritt zur Verschmelzung eines real existierenden Benutzers mit der von ihm gespielten Rolle. Spätestens hier wird klar, daß Navigation in einer VRML-Welt weit mehr bedeutet als 3D im Internet.

Kapitel 9

Audio

Der Begriff *Audio* stammt von dem lateinischen Wort audire (hören) und dient als Sammelbegriff für akustisch wahrnehmbare Signale.

Eine periodische Luftdruckschwankung wird vom menschlichen Ohr als Ton empfunden; die Höhe des Tons wird durch die Frequenz bestimmt, die Lautstärke durch die Amplitude, die Klangfarbe durch die Schwingungsform. Die Ausbreitungsgeschwindigkeit des Schalls hängt nur von den mechanischen Eigenschaften des Mediums, nicht aber von der Frequenz der Welle ab. Sie beträgt in Luft 330 m/sec = 1188 km/h.



9.1 Tonhöhe und Lautstärke

Der hörbare Bereich liegt etwa zwischen 20 Hz und 20 KHz. Der Kammerton a wurde mit 440 Hz definiert. Als musikalische Maßeinheit dient die Oktave; sie umfaßt den Abstand eines Tons bis zu dem mit doppelter bzw. halber Frequenz erzeugten Ton. Eine Oktave wird in 12 Halbtöne unterteilt (Frequenzfaktor zum Vorgänger jeweils $\sqrt[12]{2}$). Das normale Ohr kann Tonhöhendifferenzen von etwa einem zwanzigstel Halbtonschritt wahrnehmen. Eine Klaviertastatur hat 88 Tasten und erstreckt sich von 27.5 Hz bis 4186 Hz.

Die Lautstärke wird durch die Amplitude beeinflußt. Die Schallintensität wird definiert als Leistung pro Fläche, die Einheit ist also Watt/m^2. Als Schallpegel bezeichnet man den 10fachen dekadischen Logarithmus vom Verhältnis zweier Schallintensitäten. Er ist daher dimensionslos. Als Bezeichnung verwendet man das *dezibel* (dB, nach Alexander Bel). 0 dB ist der leiseste Ton, den ein Durchschnittsgehör noch wahrnehmen kann (10^{-12} Watt pro m^2). Eine Zunahme von 10 dB wird erreicht durch eine Verzehnfachung der Leistung, d.h., der von x Watt verursachte Schallpegel beträgt $10 \cdot \log \frac{x}{10^{-12}}$. Ein trainiertes Ohr kann eine Zunahme von 1 dB wahrnehmen.

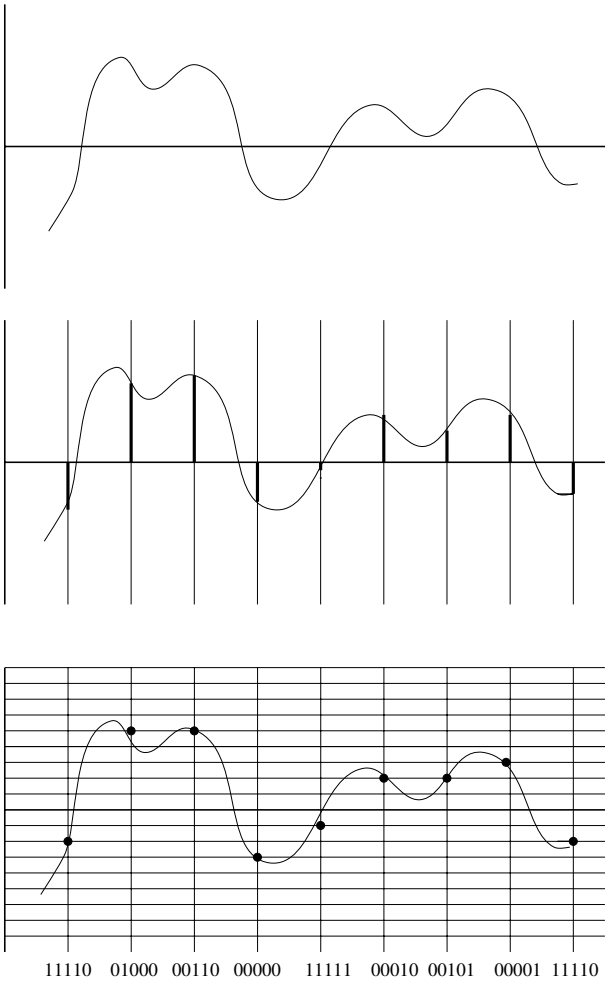
10^{-12}	0	Hörschwelle
10^{-11}	10	ruhiges Aufnahmestudio
10^{-10}	20	ruhiger Wohnraum
10^{-9}	30	ruhiges Büro
10^{-8}	40	
		gedämpfte Unterhaltung
10^{-7}	50	normales Büro
10^{-6}	60	normale Unterhaltung
		kleines Orchester
10^{-5}	70	Geschäftsstraße
10^{-4}	80	normale Fabrik
		Schwerlasttransport
10^{-3}	90	
		U-Bahn
10^{-2}	100	
		laute Rockmusik
10^{-1}	110	
		Donner
10^{0}	120	Flugzeug, Stadtbahn
		Dampfhammer
10^{1}	130	
		Schmerzgrenze
10^{2}	140	
10^{3}	150	
		Flugzeugturbine in
10^{4}	160	abgeschlossenen Räumen

Typische Lautstärkepegel, angegeben in Watt pro m^2 und in dezibel

Offenbar werden 100 dB von 10^{-2} Watt/m^2 verursacht. Eine Zunahme um 1 dB wird durch den Faktor $\sqrt[10]{10}$ erreicht. Eine Zunahme um 3 dB wird durch den Faktor $(\sqrt[10]{10})^3 = 1.995$ erreicht. Also steigt durch eine weitere Schallquelle von 100 dB der Schallpegel auf 103 dB.

9.2 Digitalisierung

Zur Digitalisierung werden die kontinuierlich-analogen Signale in gleichbleibenden Zeitabständen abgetastet und die ermittelten Werte quantisiert. Es entsteht somit eine Folge diskreter Werte, deren Qualität von der Abtastfrequenz (z.B. 10 KHz) und der Auflösung (z.B. 8 Bit für 256 Quantisierungswerte) abhängt.



kontinuierlich analoges Signal

↓ Abtastung

diskontinuierliche Folge analoger Werte

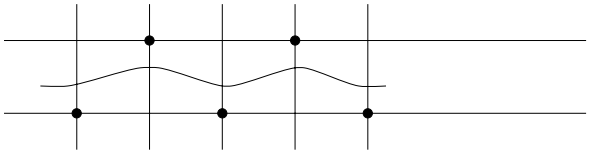
↓ Quantisierung

diskontinuierliche Folge diskreter Werte

↓ digitale Darstellung

Folge digitaler Werte als Ergebnis der Digitalisierung (in Zweierkomplement-Darstellung)

Der Quantisierungsfehler führt bei leichtem Signalschwanken durch Auf- und Abrunden zu diskreten Sprüngen, die sich als Knattern oder Quantisierungsrauschen bemerkbar machen.



Der Fehler wird in Bezug gesetzt zum Nutzsignal und beschrieben als Rauschabstand (*Signal-to-Quantization-Noise-Ratio*)

$$SNR := \frac{Abweichung\ absolut}{analoger\ Wert}.$$

Die technische Realisierung mit Hilfe eines Abtasters und Analog/Digitalwandlers nennt sich *Pulse Code Modulation* (PCM).

Lineares PCM
Die Quantisierungsschritte sind gleich groß, der Rauschabstand ist nicht konstant und macht sich bei kleinen Signalpegeln stärker bemerkbar als bei größeren.

Dynamisches PCM
Die Quantisierungsschritte werden bei wachsender Signalstärke größer (logarithmische Einteilung).

Dynamisches PCM mit Kompander
Das analoge Signal wird vor der Quantisierung gestaucht und nach der Dequantisierung wieder expandiert. Stauchungs- und Expandierungsgrad sind proportional zur Signalstärke.

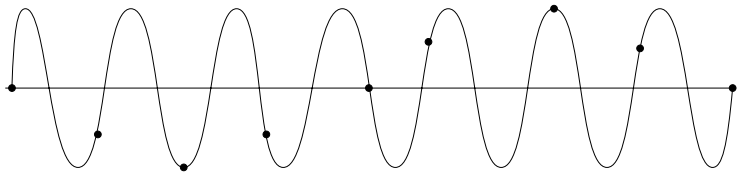
Differential PCM (DPCM)
Nach der Quantisierung werden in bestimmten Abständen Referenzpunkte gespeichert sowie die Differenzen zwischen aufeinanderfolgenden Abtastwerten. Da für die Kodierung der Differenzen weniger Bit vorgesehen sind als für die Werte selbst, versagt das System bei krassen Wechseln der Signalpegel.

Adaptive Differential PCM (ADPCM)
Durch vorausschauende Betrachtung der Abtastwerte wird festgestellt, ob man sich auf einen krassen Übergang zubewegt. Ist das der Fall, erhöht der Algorithmus schrittweise die für die Differenzkodierung verfügbare Bitanzahl und verringert sie später wieder.

Welche Auflösung und welche Abtastfrequenz sind nun für eine originalgetreue Wiedergabe erforderlich?

Der Dynamikbereich des menschlichen Ohrs beträgt etwa 100 dB. Etwa 6 dB entsprechen einem Verdoppeln der Amplitude. Bei binärer Kodierung werden also 16 Bit benötigt, um $16 \cdot 6 = 96$ dB abzudecken. Eine Auflösung von 8 Bit führt vor allem bei den leisen Tönen zu einem deutlich hörbaren Quantisierungsrauschen.

Zur Vermeidung von Aliasing-Effekten muß die Abtastfrequenz mindestens doppelt so groß sein wie die höchste vorkommende Frequenz (Abtasttheorem von Nyquist).



Beispiel für Aliasing-Effekt bei Verwendung einer Abtastfrequenz von $\frac{8}{7}$ der Originalfrequenz. Es entsteht eine verfälschte Schwingung mit einer 7-fachen Wellenlänge.

Da der hörbare Bereich sich bis 20 KHz erstreckt, ist eine Abtastfrequenz von mind. 40 KHz erforderlich. Der im *Red Book* definierte Standard für die Audio-CD (CD-DA: *Compact Disc Digital Audio*) sieht als Abtastfrequenz 44.1 KHz vor. Bei einer Auflösung von 16 Bit entsteht also bei Stereoaufnahmen im PCM-Modus eine Datenrate von $2 \times 16 \times 44100 = 1411200$ Bits/sec $\approx$ 10 MB pro Minute. Für Mono-Sprachaufnahmen im ADPCM-Modus reichen bei einer Abtastfrequenz von 18.9 KHz und 4 Bit Auflösung etwa 0.5 MB pro Minute. Für Telefonübertragungen (Frequenzbereich 200 - 3200 Hz) führt eine Abtastfrequenz von 8 KHz mit 8 Bit Auflösung bei dynamischem PCM zu $8 \times 8000 = 64$ KBit/sec $\approx$ 0.5 MB pro Minute.

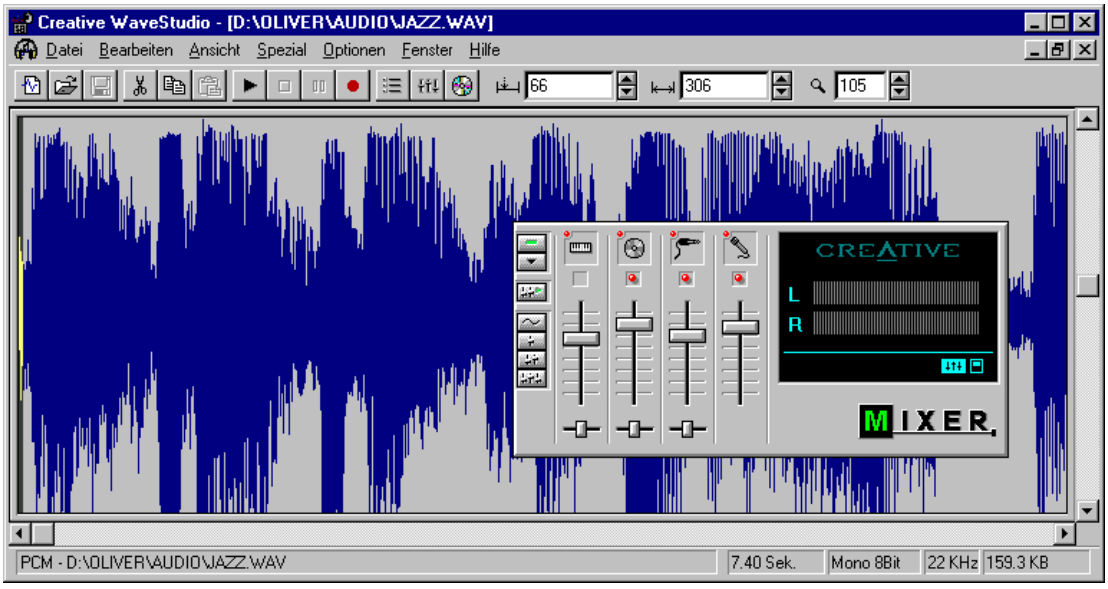
Soundkarte
Zuständig für die Analog/Digital-Wandlung und für die Digital/Analog-Wandlung ist die Soundkarte. Als Standard im PC-Bereich gilt die sogenannte *SoundBlaster-Kompatibiltät.* Die Original-*SoundBlaster*-Soundkarten wurden von der Firma Creative Labs entwickelt.

Damit ein CD-ROM-Laufwerk eine Audio-CD abspielen kann, muß ein spezieller CD-Audio-Treiber mit einem Audiokabel an der Soundkarte angeschlossen sein. Die D/A-Wandlung findet im CD-Laufwerk statt; das analoge Signal ist bereits am Kopfhörerausgang abgreifbar und wird durch die Soundkarte geschliffen, damit es, ggf. gemischt, am Line-Out- oder Speaker-Ausgang verfügbar ist.

9.3 Creative WaveStudio

Creative WaveStudio ist ein Werkzeug zur Aufnahme, Manipulation und Wiedergabe von Audiodaten.
Die charakteristischen Leistungsmerkmale lauten:

- Digitalisierung eines analogen Audiosignals,
- verschiedene Samplingfrequenzen,
- verschiedene Quantisierungsauflösungen,
- Mono/Stereo-Wahl,
- Ein- und Ausblenden,
- Invertieren,
- Echoeffekte,
- Mischen von Audioquellen.



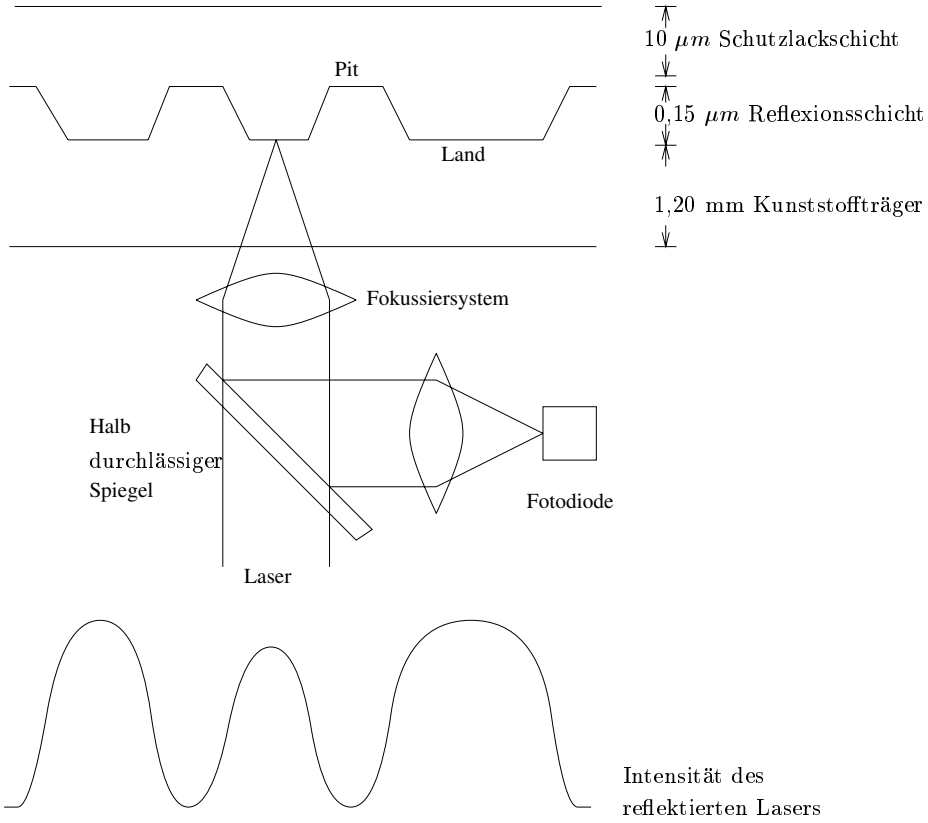
Screenshot von Audiobearbeitungswerkzeug Creative WaveStudio

Screenshot vom Wave-Plugin

9.4 Optische Speicher

Im Vergleich zu magnetischen Fest- und Wechselplatten bieten optische Datenträger eine höhere Speicherdichte zu geringeren Kosten. Die geschichtliche Entwicklung zeigt folgende Tabelle:

Jahr		Bezeichnung	Spezifikation
1982	CD-DA	Compact Disc Digital Audio	Red Book
1985	CD-ROM	Compact Disc Read Only Memory	Yellow Book
1988	CD-I	Compact Disc Interactive	Green Book



Schematische Darstellung des Abtastvorgangs

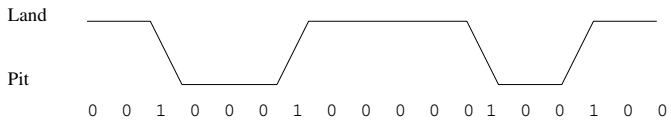
Eine optische Speicherplatte besteht aus einem durchsichtigen Kunststoffträger (1,2 μm), der auf der einen Seite mit einer Reflexionsschicht (50 nm) und einer Schutzlackschicht (10 μm) versehen ist. Die Schutzlackschicht wird mit einem Etikett bedruckt. Die Information wird durch Vertiefungen (Pit) in der Reflexionsschicht kodiert, die sich mit Nicht-Vertiefungen (Land) abwechseln. Der Höhenunterschied liegt bei etwa 0,15 μm und kann durch einen fokussierten Laserstrahl entdeckt werden: Land reflektiert stark, Pit reflektiert wenig. Die Folge von Lands und Pits bildet eine Spirale, die sich aus mehreren Spuren zusammensetzt. Die Spurbreite beträgt 0,6 μm, der Spurabstand 1,6 μm. Somit passen auf 1 mm Plattenradius mehr als 600 Spuren. Zum Vergleich: eine Diskette enthält etwa 4 Spuren pro mm. Auf einer CD mit 12 cm Durchmesser stehen etwa 4 cm Radius zur Verfügung. Dies reicht für 24000 Spuren. Die größte Spur hat einen Radius vom 6 cm, die kleinste Spur hat einen Radius von 2 cm. Die mittlere Spurlänge beträgt daher $2 \cdot \pi \cdot 4 \approx 25$ cm. Dies führt zu einer Gesamtlänge von $24000 \cdot 25$ cm $= 6$ km.

Um eine konstante Transferrate zu erreichen (*Constant Linear Velocity* = 1,3 m/sec), muß beim Lesen einer Spur die Umdrehungsgeschwindigkeit angepaßt werden: etwa 500 Umdrehungen pro Minute für die innerste Spur, etwa 200 Umdrehungen pro Minute für die äußerste Spur. Beim wahlfreien Zugriff wird daher zunächst der Lesekopf an die ungefähre Position gebracht, die Drehzahl angeglichen, dann einige Adressen gelesen, Drehzahl und Kopfposition justiert. Dies verursacht gegenüber einem *Constant-Angular-Velocity*-System, verwendet z.B. bei Magnetplatten, eine höhere Zugriffszeit.

Die binäre Information ist abgelegt durch Channelbits:

1: Es findet ein Wechsel von Land nach Pit oder von Pit nach Land statt.

0: Es findet kein Wechsel statt.

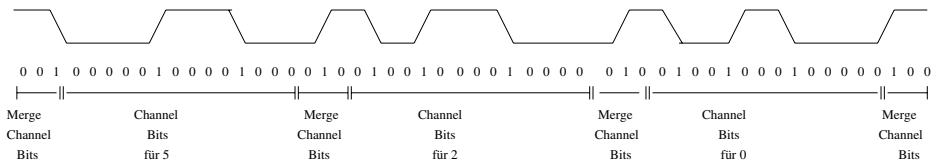


Zu schnelle Wechsel kann der Laser nicht mehr auflösen, zu lange Land- oder Pit-Folgen behindern die Synchronisation. Daher wurde festgelegt, daß mindestens 2 Pits und 2 Lands in Folge auftreten und spätestens nach 10 Pits oder Lands ein Wechsel kommen muß. Das bedeutet, daß in einer Channelbitfolge zwischen zwei Einsen mindestens zwei und höchstens zehn Nullen stehen. Dies führt zur sogenannten *Eight-to-Fourteen-Modulation* (EFM): Unter den $2^{14} = 16384$ Folgen, bestehend aus 14 Bits, befinden sich 267 zulässige Folgen, davon wurden 256 ausgesucht, die als Channelbits die acht Datenbits eines Byte kodieren.

Da beim direkten Hintereinanderfügen der Channelbit-Kodewörter die Bedingungen über minimale und maximale Pit- bzw. Landlängen verletzt werden können, werden zwischen je zwei Kodewörtern drei geeignet gewählte Merge-Channelbits eingefügt. Somit verursacht ein Datenbyte 17 Kanalbits.

Wert	Datenbits	Kodierung
0	00000000	01001000100000
1	00000001	10000100000000
2	00000010	10010000100000
3	00000011	10001000100000
4	00000100	01000100000000
5	00000101	00000100010000
6	00000110	00010000100000
7	00000111	00100100000000
8	00001000	01001001000000
9	00001001	10000001000000
10	00001010	10010001000000

Kodierung der Bytewerte 0-10



Pit-Land-Folge für die Datenbytefolge 5, 2, 0

Die Daten auf einer CD sind in sogenannten *Frames* organisiert. Ein Frame umfaßt 24 Bytes Nutzdaten (z.B. 24 Datenbytes oder 6 Stereo-Abtastpaare zu je 16 Bit). Neben den zugehörigen EFM-Kodewörtern enthält ein Frame noch Informationen zur Fehlerkorrektur und Synchronisation. Insgesamt ergeben sich 588 Bits.

Anzahl	Inhalt	Bytes	Channelbits
24	Nutzdaten	24	336
8	Fehlerkorrektur	8	112
1	Control & Display	1	14
1	Synchronisation		24
34	Mergeworte		102
			588

Inhalt eines Frame

CD-DA
Bei der Audio-CD werden 98 Frames zu einem Block zusammengefaßt. Dies ist die kleinste adressierbare Einheit und enthält 2352 Bytes Audiodaten. Alle Blöcke, die zu einem Musikstück gehören, bilden einen *Track*. Maximal 99 Tracks kann die Audio-CD enthalten.

Die 8 Bit des *Control & Display*-Byte werden aus allen Frames eines Blockes zusammengezogen und für die acht Subchannel `P, Q, R, S, T, U, V, W` mit jeweils 98 Bit Kapazität genutzt. Z.B. wird der `Q` -Channel im Vorspann zur Abspeicherung des Inhaltsverzeichnisses, im Rest der CD-DA zur Angabe der relativen Zeit innerhalb eines Tracks und der absoluten Zeitangabe auf der CD verwendet. Die Adressierung beinhaltet Minute, Sekunde und Block-Nr. Meistens hat ein Track zwei *Indexpoints*: einen für den Anfang des Tracks, einen für den Beginn der Audiodaten.

Die Fehlerkorrektur arbeitet mit dem *Cross Interleaved Real Solomon Code* und verteilt real hintereinanderliegende Audio-Bytes auf mehrere Frames eines Blockes. Auf diese Weise kann ein Loch, das mit 2 mm Durchmesser in die CD gebohrt wurde, kompensiert werden.

Bei einer Geschwindigkeit von 1,3 m/sec werden pro Sekunde 75 Blöcke gelesen. Dies führt zu einem Audiobitstrom von $75 \cdot 98 \cdot 24 \cdot 8 = 1411200$ Bit/sec. Bei 16 Bit Auflösung erlaubt dies eine Abtastfrequenz von $1411200/16 = 88200$ Hz, für Stereo also 2×44100 Hz. Da 74 Minuten Spielzeit verlangt werden, müssen $74 \cdot 60 \cdot 1411200$ Bits = 747 MByte vorhanden sein. Der Datenbitstrom berücksichtigt zusätzlich die Kontroll- und Fehlerkorrekturbytes und beträgt 1940000 Bit/sec. Der Kontrollbitstrom berücksichtigt zusätzlich den EFM-Overhead inklusive Mergebits und beträgt 4321800 Bit/sec.

CD-ROM
Die CD-DA besitzt eine Fehlerrate von 10^{-8} und ermöglicht wahlfreien Zugriff auf einzelne Tracks und Indexpoints. Für Computerdaten ist eine bessere Fehlerkorrektur und wahlfreier Zugriff mit höherer Auflösung erforderlich.

Es werden 98 Frames zu einem Block mit $98 \cdot 24 = 2352$ Bytes zusammengefaßt. Die ersten 12 Bytes werden zur Synchronisation verwendet, 3 Bytes für die Blockadresse, 1 Byte zur Mode-Bezeichnung. Mode-1-Blöcke nutzen die restlichen 2336 Bytes für Audio- oder Bilddaten. Mode-2-Blöcke verwenden von den restlichen 2336 Bytes für Computerdaten 2048 Bytes und für eine erweiterte Fehlerbehandlung 288 Bytes. Die Fehlerrate verringert sich dadurch auf 10^{-12}, d.h. 1 Fehler auf 10^{12} Bit. Bei etwa 270.000 verfügbaren Blöcken zu je 2048 Bytes bedeutet dies ein Fehler auf etwa 2000 CDs.

9.5 MPEG-Audio

Ziel ist es, die von einer Audio-CD lieferbare Tonqualität beizubehalten bei einer Datenrate von 2×192 KBit/sec. Dies entspricht einer Reduktion um den Faktor $1378/384 \approx 3.5$. Somit kann ein CD-ROM-Laufwerk mit einer Transferleistung von etwa 1.5 MBit/sec zur Wiedergabe eines Spielfilms etwa $\frac{1}{4}$ seiner Bandbreite für den komprimierten Ton, $\frac{3}{4}$ seiner Bandbreite für das komprimierte Video verwenden.
Zunächst werden die PCM-Abtastwerte mit Hilfe einer Fouriertransformation aus dem Zeitbereich in den Frequenzbereich umgesetzt. Hierzu unterteilt man die Abtastwerte in Frames mit einer definierten Anzahl von Abtastwerten. Z.B. entspricht ein Frame mit 384 Abtastwerten bei einer Abtastfrequenz von 44100 Hz einer Länge von 8.7 msec. Aus der Spektralanalyse geht hervor, welche Frequenzen in welchem Maße an dem Ausgangssignal beteiligt sind. Der Frequenzraum wird nun in 32 Subbänder à $20000/32 = 625$ Hz partitioniert. In jedem Subband werden 12 Abtastwerte ermittelt. Auf Grundlage eines psychoakustischen Modells wird nun der Maskierungseffekt von jeder Frequenz in jedem Band im Verhältnis zu jeder anderen Frequenz errechnet und für jedes Band die geeignete Auflösung eingestellt. Grundlage des psychoakustischen Modells ist die gegenseitige Überlagerung zusammengesetzter Töne. Z.B. maskiert eine Frequenz von 1000 Hz einen um mind. 18 dB leiseren Ton von 1100 Hz oder einen um mind. 45 dB leiseren Ton von 2000 Hz. In der Umgebung einer starken Frequenz ist daher ein gewisser, nicht hörbarer Grundpegel einer anderen Frequenz akzeptabel, die nun weniger Bits zur Kodierung ihrer restlichen Amplitude benötigt.
Zusätzlich verfügt der Encoder über das Wissen, daß unser Gehörorgan für hohe und niedrige Frequenzen nicht gleich sensibel ist. Der Peak liegt zwischen 2000 und 4000 Hz, wo auch die menschliche Stimme angesiedelt ist.
Zum Abschluß erfolgen Lauflängenkodierung und *Huffman*-Kodierung, und alle kodierten Spektralkomponenten werden zu Frames zusammengesetzt, deren Abfolge das kodierte Audiosignal ergibt.
Die MPEG-Audio-Layer II und III erreichen durch komplexere Implementierungen eine weitergehende Kompression bei verbesserter Performanz (d.h. Qualität pro Bitrate).
Abtastfrequenzen von 32 kHz, 44.1 kHz und 48 kHz und Datenraten zwischen 128 kBit/s und 384 kBit/s für Stereo sind vorgesehen. Bei 128 kBit/sec liefert Layer II "störende Differenzen", Layer III "wahrnehmbare Differenzen". MPEG-2 bietet als Weiterführung von MPEG-1 zusätzlich Mehrkanal- und Surround-Sound sowie die Abtastraten 16 kHz und 24 kHz.
Bei Hardware-unterstützter Realzeit-Kompression treten Verzögerungen von etwa 100 ms auf. Bei 2-Weg-Interviews gelten allerdings schon 20 ms als störend.

	Datenrate	Kompression
Layer I	384 kBit/sec	3-4
Layer II	128 kBit/sec	10-12
Layer III	64 kBit/sec	20-22

Typische Datenraten bei MPEG-1-Audio

9.6 ISO Aencode/Xaudio

ISO Aencode ist ein freeware-Produkt der International Organization for Standardization und implementiert die Layer I und II der Audio MPEG-Kompression, definiert in ISO 3-11171 Revision 1. Als Eingabeformat werden $*.wav$- oder $*.aif$-Dateien akzeptiert.
Einstellbar sind:

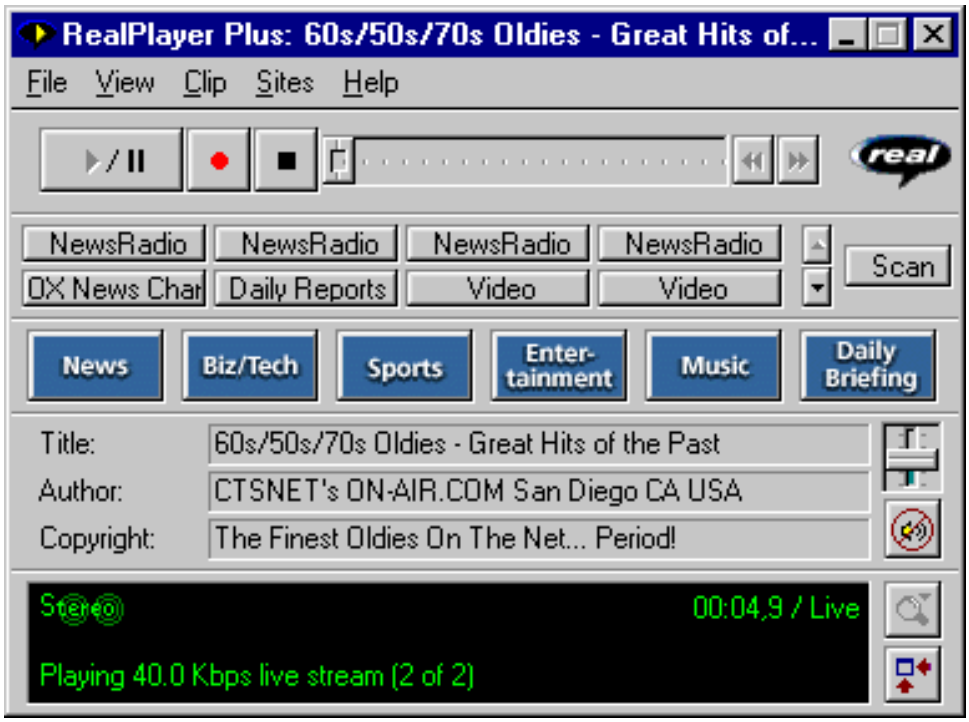
- Sampling Frequency, z.B. 44100 Hz,
- Layer, z.B. II,
- Modus, z.B. Stereo,
- psychoakustisches Modell, z.B. 2,
- Bitrate, z.B. 384 kbps.

XAudio ist ein Player zum Abspielen von MPEG-Audio-Dateien.

Screenshot vom MPEG-Audio-Player Xaudio

9.7 RealAudio

Hochkomprimierte Audiofiles verschiedener Formate sind zum Runterladen im Internet verfügbar, zum Beispiel unter der Adresse



Screenshot vom RealAudio-Plugin

.

Kapitel 10

Sprache

Dokument	Anzahl Wörter
Die zehn Gebote	300
Amerikanische Unabhängigkeitserklärung	3000
EG-Verordnung über den Import von Karamelbonbons	30000

– ohne Worte –

Von den zahlreichen Aspekten der Sprachverarbeitung soll hier nur das Worterkennungsproblem behandelt werden. D.h., ein gesprochener Satz von mehreren Einzelwörtern soll in eine ASCII-Repräsentation überführt werden. Es geht also nicht um die Analyse der Bedeutung des Satzes, wohl aber um die richtige Schreibweise der beteiligten Wörter.

13	ab	29	diesen	19	große	17	numerischen	12	transputernetz
132	aber	100	dieser	30	großen	29	nun	52	über
13	ablaufen	59	dieses	36	haben	144	nur	54	um
21	abschnitt	15	dimension	12	häufig	18	oben	11	unabhängig
27	adaption	17	direkt	17	hardware	46	oder	423	und
18	adaptionsschritt	29	dort	50	hat	21	ohne	16	ungleichgewichte
25	algorithmen	16	drei	10	heute	12	operationen	20	unstrukturierten
50	algorithmus	122	durch	129	hier	13	optimal	14	unter
106	alle	49	durchgeführt	11	hilfe	11	optimale	10	unterschiede
19	allen	12	dynamische	172	im	67	optimierung	13	untersuchen
110	als	11	dynamischen	51	immer	13	optimierungsschritte	11	varianten
23	also	17	ebenfalls	12	implementiert	26	parallel	61	verfahren
20	am	15	effekte	341	in	25	parallele	23	verfahrens
85	an	13	effiziente	24	indem	70	parallelen	12	verfeinert
18	andere	20	effizienz	17	innerhalb	27	parallelisierung	12	verfeinerung
50	anderen	11	effizienzen	15	insbesondere	14	parallelität	26	verfügung
34	anzahl	289	ein	401	ist	19	parix	10	verhalten
10	arbeit	308	eine	10	jede	11	periodische	10	verschieben
11	arbeiten	115	einem	56	jedem	20	periodischen	24	verschiedene
14	arbeitet	70	einen	14	jeden	14	phase	29	verschiedenen
11	art	130	einer	32	jeder	13	phasen	15	version
141	auch	57	eines	13	jedes	13	praktisch	20	verteilt
256	auf	14	einfach	40	jetzt	58	problem	19	verteilung
11	aufgrund	17	einfache	32	jeweils	63	probleme	15	verwenden
64	aufteilung	39	einige	146	kann	15	problemen	32	verwendet
84	aus	15	einigen	20	kapitel	10	problems	11	verwendete
15	austausch	14	einmal	12	kein	29	programm	31	verwendeten
11	bearbeitet	19	einsatz	37	keine	10	programme	17	verwendung
10	behandelt	11	einzelnen	14	klar	20	programms	34	viele
228	bei	92	element	11	kleine	104	prozessor	10	vielen
42	beiden	152	elemente	14	kleinen	138	prozessoren	19	vier
22	beim	40	elementen	10	kleiner	31	prozessorzahlen	13	virtuellen
23	benötigt	14	elements	145	knoten	21	punkt	18	völlig
11	berechnen	13	enthalten	82	können	18	punkte	17	vollständig
28	berechnet	14	entsprechend	24	kommunikation	18	ränder	19	vom
23	berechnung	22	entsprechenden	25	kommunikationen	14	rand	219	von
52	berechnungen	10	entwickelt	34	koordinaten	21	randbedingungen	10	vor
14	bereich	14	er	11	kurz	16	rechenzeit	18	vorhanden
12	berücksichtigt	62	ergeben	12	läßt	10	rechner	10	vorigen
10	beschleunigung	11	ergebnis	14	lassen	21	relativ	16	während
15	beschränkt	17	ergebnisse	22	last	37	schon	14	war
21	beschrieben	79	ergibt	11	lastverteilung	28	schritt	21	was
13	beschriebenen	12	erhalten	27	laufzeit	18	schritte	25	weitere
18	besonders	24	erreicht	15	laufzeiten	14	sehen	22	weiteren
18	besteht	12	erst	15	liegen	76	sehr	12	weiterer
32	bestimmt	15	ersten	20	liegt	40	sein	13	wenige
11	bestimmung	18	erweiterungen	19	links	11	seine	74	wenn
13	beteiligt	11	erzeugt	13	listen	13	seite	418	werden
11	beteiligten	172	es	47	lösung	18	selbst	15	wert
21	betrachten	17	etwa	10	lösungen	30	sequentiellen	18	werte
10	betrachtet	12	exakt	18	lokal	265	sich	16	wesentlichen
10	betriebssystem	14	fällen	25	lokale	25	sie	74	wie
51	bild	55	fall	45	lokalen	167	sind	41	wieder
32	bis	15	fast	92	man	12	sinnvoll	105	wir
19	bzw	11	fein	18	massiv	161	so	229	wird
99	da	10	feste	10	maximal	28	solche	25	wo
76	dabei	10	finiten	49	mehr	18	solchen	23	wobei
12	dagegen	11	folgende	14	mehrere	10	solcher	21	wollen
35	daher	37	folgenden	10	meist	11	solches	15	workstation
48	damit	15	frage	21	messungen	24	soll	26	wurde
15	danach	23	führen	16	methode	17	sollen	43	wurden
144	dann	23	führt	232	mit	17	sollte	12	zahl
11	daraus	309	für	30	möglich	18	sondern	12	zeigt
10	dargestellt	13	funktionen	27	möglichst	24	speedup	10	zeilen
183	das	10	ganz	67	müssen	22	speedups	24	zeit
286	daß	12	gebiets	70	muß	16	speicher	10	zeiten
40	daten	12	geeignet	50	nach	16	spezielle	22	zeitschritt
12	datenstrukturen	10	geeigneten	25	natürlich	15	speziellen	14	zeitschritte
47	dazu	10	gegen	46	netz	13	startverteilung	275	zu
95	dem	13	genau	14	netzaufteilung	17	strategie	16	zugehörigen
238	den	12	geometrie	16	netzdichte	17	strömungen	28	zum
38	denen	10	gerade	37	netze	12	strömungsgrößen	45	zunächst
811	der	53	gibt	40	netzen	17	struktur	73	zur
177	des	13	gilt	27	netzes	18	system	20	zusätzlich
30	deutlich	12	gleichungen	24	neue	13	systeme	14	zusätzliche
1159	die	17	globale	26	neuen	15	systemen	12	zusätzlichen
95	dies	15	globalen	164	nicht	12	tatsache	12	zwar
177	diese	10	grenzen	60	noch	12	teil	77	zwei
11	dieselben	15	größe	25	notwendig	16	teile	11	zweite
50	diesem	12	größen	26	notwendigen	27	transputer	41	zwischen

Die 400 häufigsten Worte aus einer naturwissenschaftlichen Dissertation.
Die Stichprobe hat einen Umfang von 25941 Worten. Es gibt 3567 verschiedene Worte, 1728 kommen einmal vor, 400 kommen mindestens 10 mal vor.

Der typische Sprachumfang eines Engländers beträgt 800 Wörter, eines Deutschen 4000 Wörter. Goethe beherrschte etwa 24000 Wörter. Englische Verben haben etwa 4 Flexionen, deutsche etwa 10:

speak	spreche
speaks	sprichst
spoke	spricht
spoken	sprechen
	sprecht
	sprach
	sprachst
	sprachen
	spracht
	spräche
	sprächest
	sprächen
	sprächet

Durch Komposita und Derivationen entstehen im Deutschen neue Wörter

```
Elbe + Mündung    =  Elbmündung
Hochzeit + Torte  =  Hochzeitstorte
hin + fahren      =  hinfahren
```

Insgesamt führt dies im deutschen Sprachraum zu etwa 1 000 000 Wortformen.

Eine zusätzliche Schwierigkeit verursachen die Homophone, z.B. Meer/mehr; floh/Floh. Ihre korrekte Schreibweise läßt sich nur kontextbezogen ermitteln:

```
Der junge Junge fiel viel und fällt noch immer viel auf dem Feld.
                        Der gefangene Floh.
                        Der Gefangene floh.
```

Die Worterkennung läuft in 3 Phasen ab:

1. Generieren von Merkmalsvektoren mit Fourieranalyse.
2. Generieren von Lautschriftkandidaten mit Hidden-Markow-Modellen.
3. Generieren von Wortkandidaten mit Trigrammen.

10.1 Akustische Vorverarbeitung

Die Erkennung eines Sprachsignals beginnt mit der akustischen Vorverarbeitung. Etwa alle 10 ms wird das analoge Eingangssignal einer Fourieranalyse unterworfen und die beteiligten Frequenzen ermittelt. Die Werte zweier aufeinander folgender Kurzzeitspektren werden einer Diskriminanzanalyse unterworfen und ergeben dann einen sogenannten *Merkmalsvektor*.

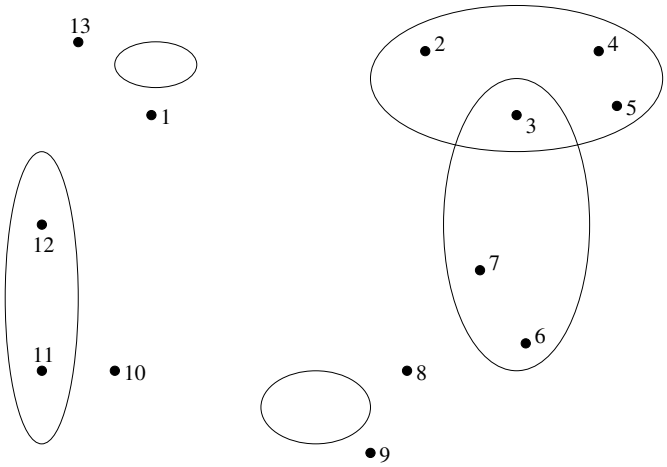
Die kleinste lautsprachliche Einheit nennt man *Phonem*. Im Deutschen gibt es etwa 40 Phoneme (Konsonanten, kurze und lange Vokale und Diphtonge):

b, p, d, t, f, w, g, k, l, m, n, r, s, ß, sch, z, x
a:, a, e:, e, i:, i, o:, o, u:, u,
ä:, ä, ö, ö:, ü, ü:, ai, au, ui.

Die während der Artikulation eines Phonems erzeugten Merkmalsvektoren können durch eine Wolke in einem hochdimensionalen Raum charakterisiert werden.

Form und Lage einer Phonem-Wolke ist sprecherabhängig und wird in einer mehrstündigen Trainingsphase ermittelt. Aus Gründen der Rechenökonomie beschreibt man die Wolken als Kugeln oder Ellipsoide. Ggf. wird der zuständige Teilraum durch mehrere Standardwolken angenähert.

Genauer: Je näher ein Merkmalsvektor dem Mittelpunkt eines Ellipsoids liegt, desto größer ist die Wahrscheinlichkeit, daß er zu dem entsprechenden Phonem gehört. Auf die Fläche des Ellipsoids fallen im Mittel 50 % der Realisierungen des entsprechenden Phonems. Ein Punkt außerhalb kann durchaus zu diesem Phonem gehören, aber es ist recht unwahrscheinlich.



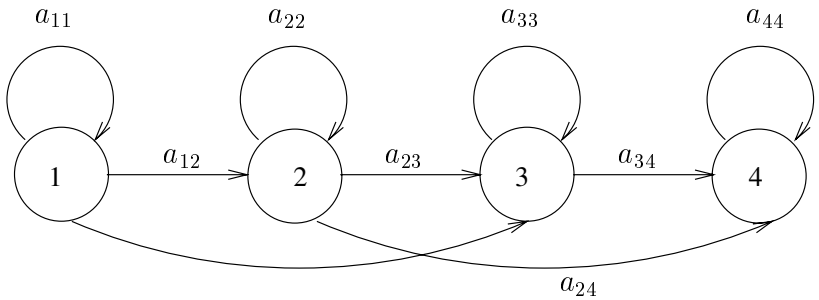
13 Merkmalsvektoren und ihre Beziehung zu 5 Phonem-Wolken

10.2 Hidden-Markow-Ketten

Gesteuert von der in der akustischen Vorverarbeitung entstandenen Folge von Merkmalsvektoren hüpft nun ein Punkt im Merkmalsraum von Wolke zu Wolke. Bei der Rekonstruktion der mutmaßlichen Lautschrift treten zwei Probleme auf:

1. Die Wolken überlappen sich.
2. Die zeitliche Abfolge der an einem Wort beteiligten Merkmalsvektoren ist nicht eindeutig festgelegt, sondern nur die ungefähre Reihenfolge.

Zur Lösung dieses Problems wird die Folge der möglichen Merkmalsvektoren für ein festes Wort w als Markow-Kette beschrieben. Die Zustände enthalten Wahrscheinlichkeitsverteilungen von Merkmalsvektoren, die Kanten beschreiben mögliche Übergänge, gewichtet mit ihrer Übergangswahrscheinlichkeit.
Genauer: $p(\overline{x}|j)$ ist die Erzeugungswahrscheinlichkeit für Merkmalsvektor $\overline{x}$ im Zustand $1 \leq j \leq r$; a_{ij} ist die Übergangswahrscheinlichkeit vom Zustand i zum Zustand j.



Markow-Kette für ein Wort mit 4 Zuständen

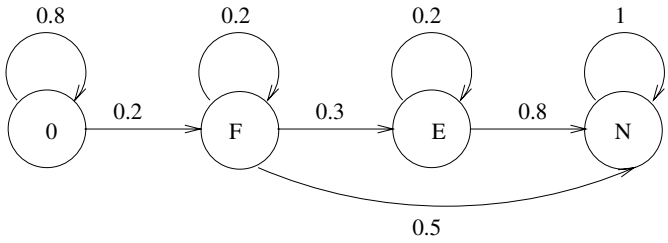
Zu einer beobachteten Folge von Merkmalsvektoren $\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_T$ ist nun die Wahrscheinlichkeit P zu bestimmen, mit der diese Folge von der zu W gehörigen Markow-Kette erzeugt werden kann. P errechnet sich als Summe über alle gewichteten Wege der Länge T, beginnend beim Startzustand 1, endend beim Endzustand r.
Sei $\alpha_t(j)$ = Wahrscheinlichkeit, daß bis zum Zeitpunkt t unter Erzeugung des korrekten Merkmalsverlaufs $\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_t$ der Zustand j erreicht wurde.
Offenbar

$$\alpha_1(j) = \begin{cases} p(\overline{x}_1|1) & \text{für } j = 1 \\ 0 & \text{sonst} \end{cases}$$

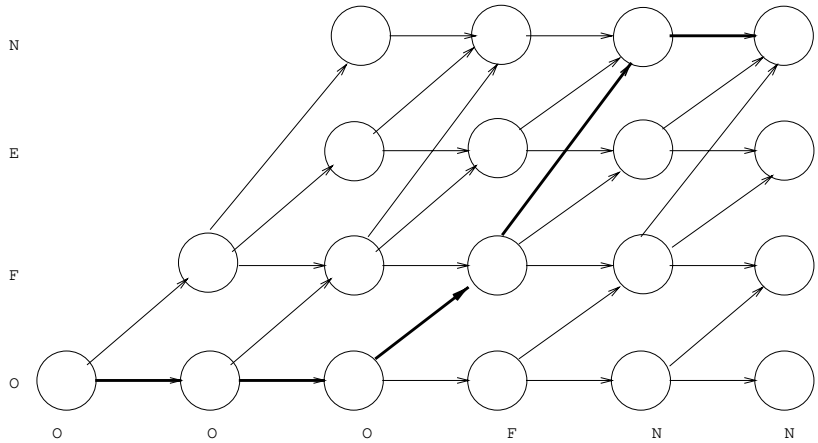
$$\alpha_{t+1}(j) = (\sum_{i=1}^{r} \alpha_t(i) \cdot a_{ij}) \cdot p(\overline{x}_t|j) \text{ für } j = 1, \ldots, r$$

$$P = \alpha_T(r)$$

Als Vereinfachung kann sich die Rechnung auf die Bestimmung der optimalen Kombination beschränken (*Viterbi*-Algorithmus). Dies ist dann sinnvoll, wenn nur eine einzige Kombination zur gesamten Wahrscheinlichkeit wesentlich beiträgt. Durch Logarithmieren der Wahrscheinlichkeiten entsteht ein kürzeste-Wege-Problem.



Markow-Kette für das Wort `OFEN`
Die Erzeugungswahrscheinlichkeiten seien
jeweils 1 für das "richtige" Phonem zum "richtigen" Zeitpunkt.



Kürzester Weg für beobachtete Phonemfolge `OOOFNN`

Für alle im System gespeicherten Markow-Ketten wird die Pfadsuche gleichzeitig durchgeführt und liefert somit eine Liste von Lautschriftkandidaten, gewichtet mit ihrer Erzeugungswahrscheinlichkeit.

10.3 Trigramme

Welcher von den wahrscheinlichsten Lautschriftkandidaten der richtige ist und wie seine Schreibweise lautet, wird kontextbezogen ermittelt, da bei gegebenem Sprachsignal die Wahrscheinlichkeit für eine Wortfolge proportional zum Produkt aus Synthese- und Grundwahrscheinlichkeit ist. Z.B. wird in einem medizinischen Befund, diktiert von einem Zahnarzt, das Wort *Zähne* eine deutlich höhere Grundwahrscheinlichkeit besitzen als das Wort *Sehne*, so daß selbst bei recht hoher Synthesewahrscheinlichkeit für *Sehne* die beobachtete Merkmalsvektorfolge auf *Zähne* abgebildet werden sollte. Hierzu werden in einer Datenbank Häufigkeitsbeobachtungen von Dreiwortfolgen (*Trigrammen*) gespeichert. Grundlage sind vom Anwender gelieferte Textproben. Die Häufigkeiten von nicht beobachteten Dreiwortfolgen müssen geschätzt werden.

10.4 IBM ViaVoice

IBM ViaVoice ist eine sprecheradaptives Worterkennungssystem für kontinuierlich gesprochene Sprache, welches außer einer Soundblaster-kompatiblen Soundkarte keine Spezialhardware erfordert. Zum Lieferumfang gehört ein Universal-Wortschatz von 20 000 Wörtern, der sich bei der Benutzung ständig den Sprechgewohnheiten des Benutzers anpaßt. Nach einer etwa einstündigen Trainingsphase ist das System auf einen bestimmten Sprecher eingestellt.

10.5 Automatische Sprachübersetzung

```
Sehr geehrte Damen und Herren,
bezugnehmend auf unser Schreiben vom 27. September dieses Monats weisen wir
nochmals darauf hin, daß Ihre Außenstände bei unserem Unternehmen nunmehr den
Betrag von 4.000 DM übersteigen.
Wir fordern Sie daher unwiderruflich auf, den offenen Betrag auf unser Konto
4711 bei der Hamburger Sparkasse zu überweisen.
Hochachtungsvoll
Willi Wacker
```

Online Translation von Systranet (http://www.systranet.com) für 0.86 US-$:

```
Ladies and Gentlemen,
referring to our letter from 27 September of this monthly we point out again
that your accounts receivable exceed the amount of 4.000 DM with our
enterprise now.
We request you irrevocablly to transfer the open amount to our account 4711
with that to Hamburg savings bank.
Faithfully
Willi Wacker
```

Kapitel 11

Musik

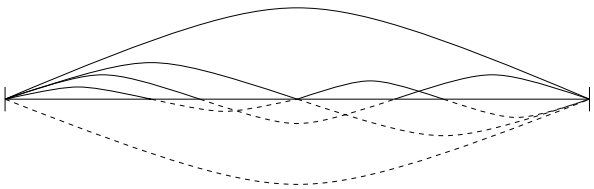
Periodische Luftdruckschwankungen werden vom menschlichen Ohr
als Ton empfunden.

Hierbei wird

> die Tonhöhe durch die Schwingungsfrequenz,
> die Lautstärke durch die Schwingungsamplitude,
> die Klangfarbe durch die Schwingungszusammensetzung

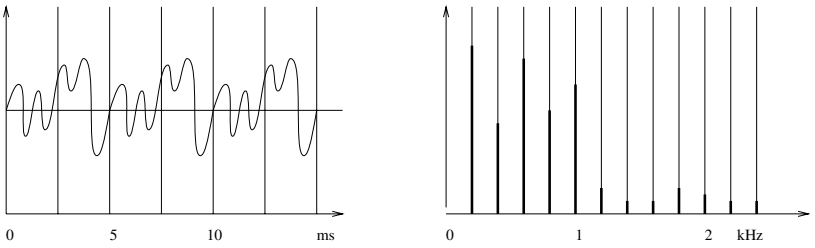
geregelt.

2-dimensionale, natürliche Schwingungssysteme (z.B. eingespannte Saite, eingesperrte Luftsäule) erzeugen zur Grundfrequenz stets ihre ganzzahligen Vielfachen, genannt *Obertöne*, mit unterschiedlicher Intensität.



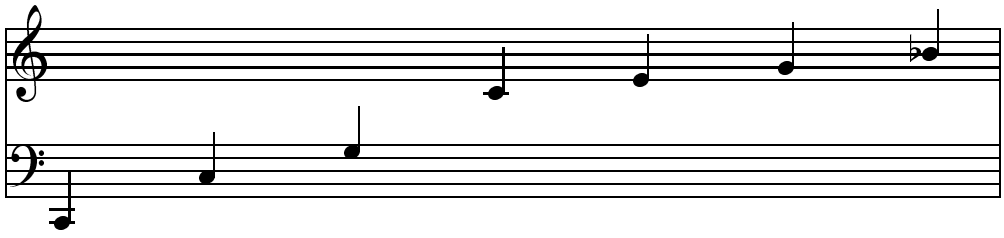
1-, 2-, 3-, 4-fache Grundfrequenz einer eingespannten Saite

Somit läßt sich eine Klangfarbe beschreiben durch ein Frequenzspektrum, welches die Amplituden der beteiligten Frequenzen enthält.



Klang, 200 Hz
period.-harm. Aufbau

Die Abstände zwischen den Obertönen definieren gewisse natürliche Tonintervalle.



1	:	2	:	3	:	4	:	5	:	6	:	7
C		c		g		c'		e'		g'		b'

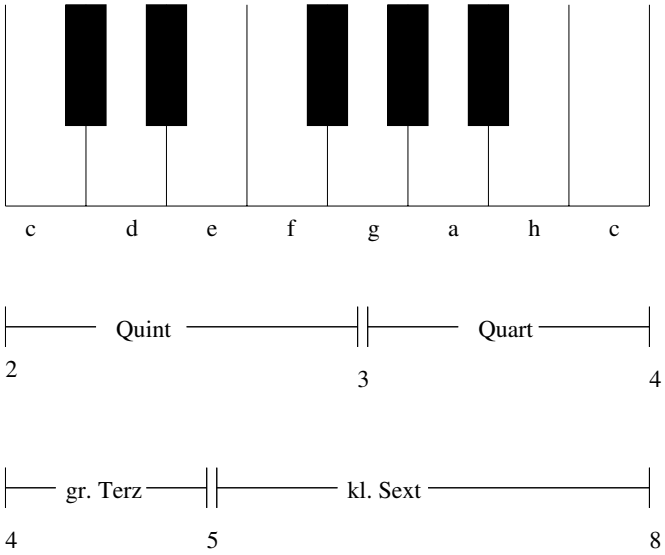
⊢ Oktave ⊣ ⊢Quinte ⊣ ⊢ Quarte⊣ ⊢gr. Terz⊣ ⊢kl. Terz⊣

Intervalle zwischen den Obertönen

11.1 Tonsysteme

Unser heute verwendetes Tonsystem berücksichtigt die Tatsache, daß zwei Töne mit dem Frequenzverhältnis 1 : 2 als gleich empfunden werden. Zwischen ihnen liegt eine Oktave. Dieses Intervall ist in 12 Halbtöne geteilt, jeweils mit dem Frequenzfaktor $\sqrt[12]{2}$. Entstanden sind diese 12 Halbtöne aus dem Versuch, zueinander passende Töne zu einem Tonsystem zusammenzufassen.

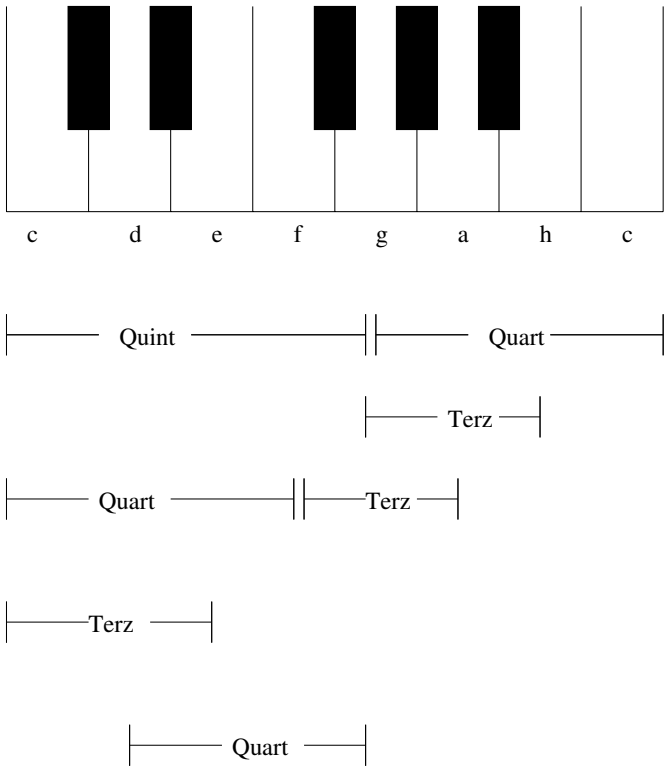
Töne werden als konsonant (zusammenklingend) empfunden, wenn ein oder mehrere ihrer Obertöne (bis zum 8. Partialton) zusammenfallen. Dies bedeutet, daß sich ihr Frequenzverhältnis durch einen Bruch mit kleinem Zähler und Nenner beschreiben läßt. Andernfalls werden sie als dissonant bezeichnet.



Ausgehend von einem Grundton lassen sich nun weitere Töne zu einem Tonsystem zusammenstellen.

Pentatonik
Es werden so lange Quinten geschichtet, d.h. Frequenzen mit Faktor 3/2 hinzugefügt, bis der Abstand weniger als 1 Ganzton beträgt. Beginnend bei `c` entsteht `c - g - d - a - e`. Verschoben um einen halben Ton ergibt dies genau die schwarzen Tasten eines Klaviers. Die Töne haben die Abstände 1, 1 1/2, 1, 1.

Diatonik
Es werden innerhalb einer Oktave Quinten, Quarten und große Terzen kombiniert. Beginnend bei `c` ergibt dies genau die weißen Tasten eines Klaviers. Die Töne haben die Abstände 1, 1, 1/2, 1, 1, 1, 1/2.

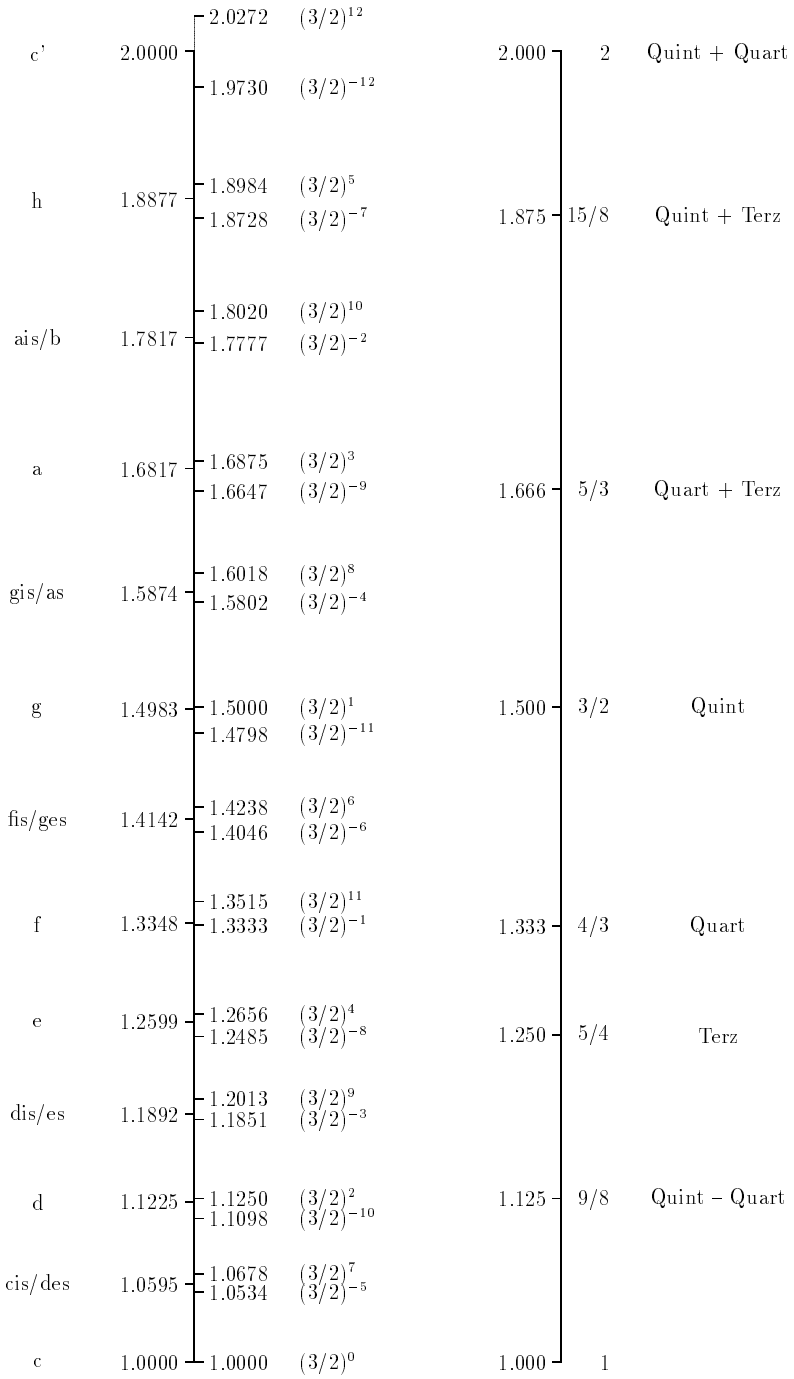


Dieselben Töne entstehen, wenn, ausgehend vom `d`, jeweils 3 Quinten nach oben und nach unten gegangen wird.

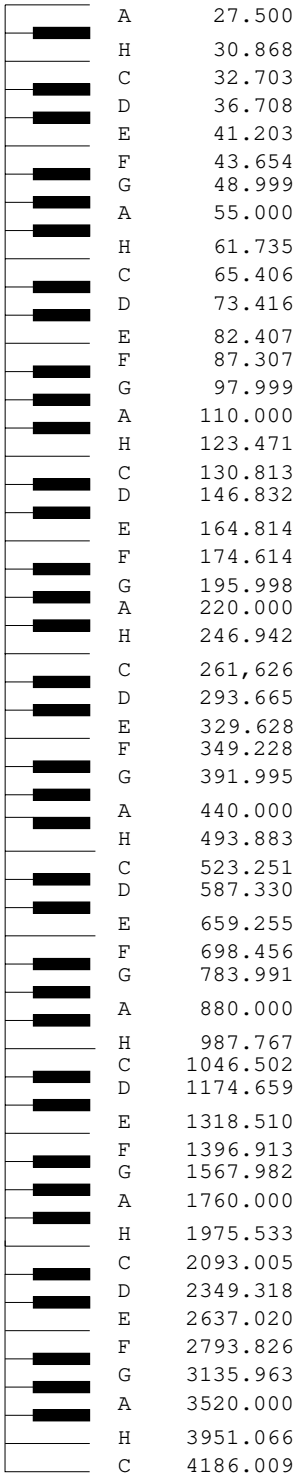
Chromatik
Es werden so lange Quinten geschichtet, bis der Abstand weniger als ein Halbton beträgt. Dies ist nach 12 Quinten der Fall, denn

$$\left(\frac{3}{2}\right)^{12} = 129.74 \approx 128 = \left(\frac{2}{1}\right)^{7}.$$

Der Abstandsfaktor beträgt somit 1.0135, das ist weniger als ein Viertel Halbton. Es entstehen innerhalb einer Oktave 24 verschiedene Töne, von denen je zwei dicht beieinanderliegen. Durch Zusammenfassung werden daraus 12 Halbtöne mit einem Frequenzfaktorabstand von $\sqrt[12]{2}$. Dies sind genau die schwarzen und weißen Tasten eines Klaviers.



Quintenzirkel, beginnend bei c
12 reine Quinten aufwärts und abwärts



Schwingungsfrequenzen eines 88-Tasten-Klaviers

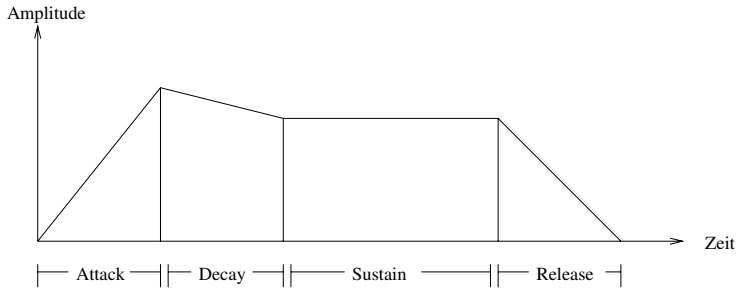
11.2 Tongenerator

Der typische Klang eines Musikinstruments ergibt sich aus den Amplituden der beteiligten Obertöne und aus dem charakteristischen zeitlichen Ablauf beim Erzeugen und Abklingen eines Tons.

Zunächst liefert der *Oszillator* das Rohmaterial, d.h. eine Sammlung von reichhaltig zusammengesetzten Schwingungen. Hieraus formt der Filter den gewünschten Instrumentenklang durch Reduktion einiger Obertöne. Schließlich liefert der Hüllkurvengenerator an den Verstärker den dynamischen Lautstärkeverlauf.

Eine Hüllkurve ist meistens aus vier Teilen zusammengesetzt:

- *Attack*: Einschwingzeit,
- *Decay*: Herabsinken zum Haltepegel,
- *Sustain*: Haltepegel,
- *Release*: Ausschwingzeit.



11.3 MIDI

Die Sprache MIDI (*Musical Instrument Digital Interface*) entstand 1982 als Ergebnis einer Kooperation der Gerätehersteller Sequential Circuits, Roland, Yamaha, Kory und Kawaii und war ursprünglich konzipiert als Kommunikationssystem zur Ansteuerung elektronischer Musikinstrumente. Mit der Verbreitung von Personal-Computern kamen weitere Komponenten hinzu (1987 *Midi Time Code*, 1988 *Midi Files*). Inzwischen ist *General Midi* ein weitverbreiteter Standard zur Spezifikation, Manipulation, Übertragung und Speicherung von elektronisch erzeugter Musik. In Kombination mit einem PC reicht ein sogenanntes *Masterkeyboard*, welches über ein 5-poliges Kabel die Beschreibung des Musikstückes an die Soundkarte schickt, wo sie dann von einem Synthesizer, ggf. mit Unterstützung von digital abgelegten Klangsamples, in Töne umgesetzt wird. Midinachrichten sind Byte-orientiert und werden über eine serielle Schnittstelle mit 31250 Baud übertragen. Es wird unterschieden zwischen

Status-Byte	1	B	B	B	K	K	K	K
Data-Byte	0							

Status-Bytes kündigen eine Aktion an und gliedern sich in *Kanal-* und *Systembefehle*. Kanalbefehle (`BBB`) beziehen sich auf einen der insgesamt 16 Kanäle (`KKKK`), mit denen jeweils ein angeschlossenes Instrument assoziiert ist:

`000`	`Note off`	schaltet Ton aus
`001`	`Note on`	schaltet Ton an
`010`	`Polyphonic Aftertouch`	Anschlagdruck für eine Taste
`011`	`Control Change`	Realzeitmeldung, z.B. Lautstärke
`100`	`Program Change`	Instrument/Klangfarbe
`101`	`Channel Aftertouch`	Anschlagdruck für alle Tasten
`110`	`Pitch Bend`	Auslenkung Tonhöhenrad
`111`	`Systembefehl`	z.B. zur Synchronisation

Systembefehle werden über die letzten vier Bits weiter spezifiziert.

Einem Statusbyte folgen ggf. mehrere Datenbytes. Z.B. gehören zum `Note-On` -Befehl zwei Parameter: Notennummer und Anschlaggeschwindigkeit. Mit 128 Werten können mehr als 10 Oktaven abgedeckt werden, eine Klaviertastatur mit 88 Tasten von Subkontra A bis zum fünfgestrichenen C belegt das Intervall 33 bis 120. Zum `Note-Off` -Befehl gehören wiederum Notennummer und Loslaßgeschwindigkeit.

Um den zeitlichen Aspekt beim Abspielen einer Midi-Datei zu rekonstruieren, wird die Zeitachse zu sogenannten *Tics* diskretisiert. Z.B. beträgt beim Sequenzer Cubasis die zeitliche Auflösung 384 Tics pro Viertelnote.
Beim Abspeichern einer Notensequenz werden die Intervalle zwischen den Events als sogenannte *Delta Times* abgespeichert. Hierzu werden bis zu 4 Bytes verwendet.

Piano	1	Piano 1 (Acoustic Grand)	Reed	65	Soprano Sax
	2	Piano 2 (Bright Acoustic)		66	Alto Sax
	3	Piano 3 (Electric Grand)		67	Tenor Sax
	4	Honky-Tonk Piano		68	Baritone Sax
	5	E-Piano 2		69	Oboe
	6	E-Piano 2		70	English Horn
	7	Harpsicord		71	Bassoon
	8	Clavinett		72	Clarinet
Chrom. Percussion	9	Celesta	Pipe	73	Piccolo
	10	Glockenspiel		74	Flute
	11	Musicbox		75	Recorder
	12	Vibraphone		76	Pan Flute
	13	Marimba		77	Bottle Blow
	14	Xylophone		78	Shakuhachi
	15	Tubular Bell		79	Whistle
	16	Santur		80	Ocarina
Organ	17	Organ 1	Synth. lead	81	Square Wave
	18	Organ 2		82	Saw Wave
	19	Organ 3		83	Syn. Calliope
	20	Church Org. 1		84	Chiffer Lead
	21	Reed Organ		85	Charang
	22	Accordion Fr		86	Solo Vox
	23	Harmonica		87	5th Saw Wave
	24	Bandeon		88	Bass & Lead
Guitar	25	Nylon-str. Gt.	Synth. pad	89	Fantasia
	26	Steel-str. Gt.		90	Warm Pad
	27	Jazz Gt.		91	Polysynth
	28	Clean Gt.		92	Space Voice
	29	Muted Gt.		93	Bowed Glass
	30	Overdrive Gt.		94	Metal Pad
	31	Distortion Gt.		95	Halo Pad
	32	Gt. Harmonics		96	Sweep Pad
Bass	33	Acoustic Bs.	Synth. SFX	97	Ice Rain
	34	Fingered Bs.		98	Soundtrack
	35	Picked Bs.		99	Crystal
	36	Fretless Bs.		100	Atmosphere
	37	Slap Bass 1		101	Brightness
	38	Slap Bass 2		102	Goblin
	39	Synth. Bass 1		103	Echo Drops
	40	Synth. Bass 2		104	Star Theme
Strings/orchestra	41	Violin	Ethnic	105	Sitar
	42	Viola		106	Banjo
	43	Cello		107	Shamisen
	44	Contrabass		108	Koto
	45	Tremolo Str		109	Kalima
	46	PizzicatoStr		110	Bag Pipe
	47	Harp		111	Fiddle
	48	Timpani		112	Shannai
Ensemble	49	Strings	Percussive	113	Tinkle Bell
	50	Slow Strings		114	Agogo
	51	Syn. Strings1		115	Steel Drums
	52	Syn. Strings2		116	Woodblock
	53	Choir Aahs		117	Taiko
	54	Voice Oohs		118	Melo Tom 1
	55	Syn. Vox		119	Synth. Drum
	56	OrchestraHit		120	Reverse Cym.
Brass	57	Trumpet	SFX	121	Gt. FretNoise
	58	Trombone		122	Breath Noise
	59	Tuba		123	Seashore
	60	Muted Trumpet		124	Bird
	61	French Horn		125	Telephone 1
	62	Brass 1		126	Helicopter
	63	Synth. Brass1		127	Applause
	64	Synth. Brass2		128	Gun Shot

Voreinstellung für General Midi

Taste	Noten-Nr.	Instrument
B1	35	Bassdrum
C2	36	Bassdrum 1
C#2	37	Rimshot
D2	38	Snare Drum 1
D#2	39	Hand Clap
E2	40	Snare Drum 2
F2	41	Low Tom 2
F#2	42	Closed Hi-Hat
G2	43	Low Tom 1
G#2	44	Pedal Hi-Hat
A2	45	Mid Tom 2
A#3	46	Open Hi-Hat
B2	47	Mid Tom 1
C3	48	High Tom 2
C#3	49	Vrash Cymbal
D3	50	High Tom 1
D#3	51	Ride Cymbal 1
E3	52	Chinese Cymbal
F3	53	Ride Bell
F#3	54	Tambourin
G3	55	Splash Cymbal
G#3	56	Cowbell
A3	57	Crash Cymbal 2
A#3	58	Vibra-Slap
B3	59	Ride Cymbal 2
C4	60	High Bongo
C#4	61	Low Bongo
D4	62	Mute High Conga
D#4	63	Open High Conga
E4	64	Low Conga
F4	65	High Timbale
F#4	66	Low Timbale
G4	67	High Agogo
G	68	Low Agogo
A4	69	Cabasa
A#4	70	Maracas
B4	71	Short Hi Whistle
C5	72	Long Low Whistle
C#5	73	Short Guiro
D5	74	Long Guiro
D#5	75	Claves
E5	76	High Wood Block
F5	77	Low Wood Block
F#5	78	Mute Cuica
G5	79	Open Cuica
G#5	80	Mute Triangel
A5	81	Open Triangel

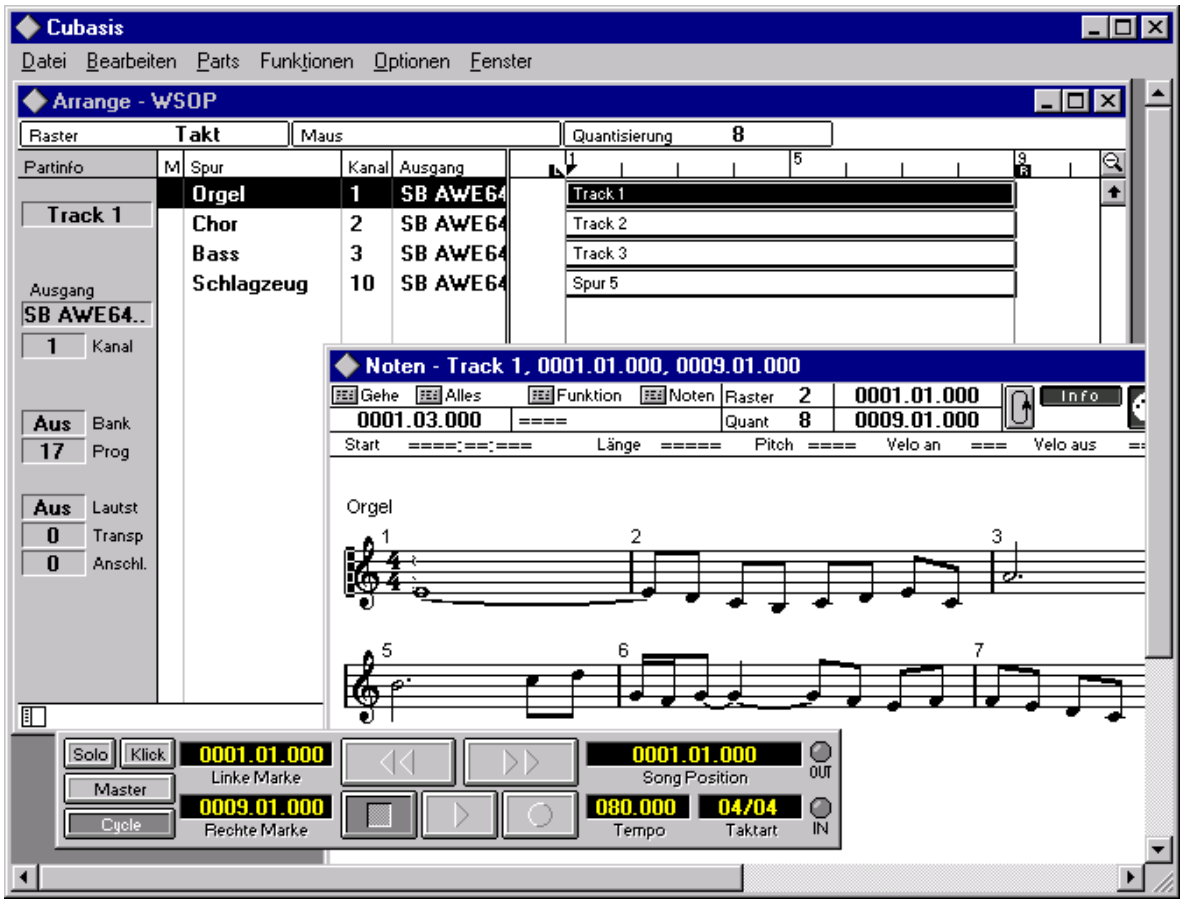
Schlagzeugeinstellung für Kanal 10

11.4 Roland Masterkeyboard PC-200

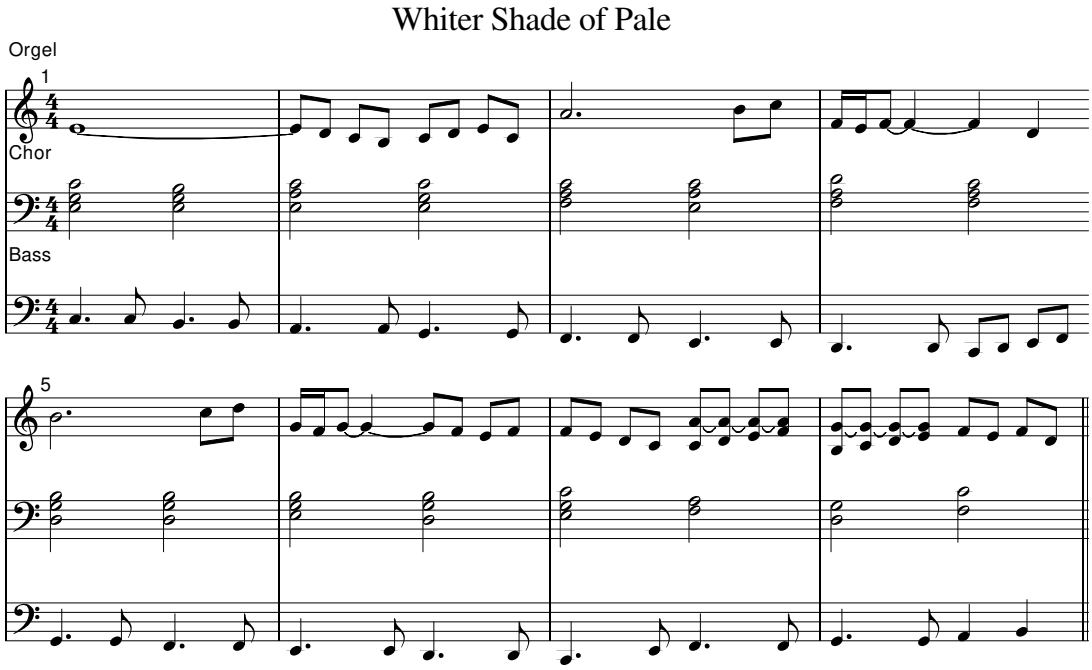
Das *Roland Masterkeyboard PC-200* verfügt über 49 anschlagsdynamische, polyphon spielbare Tasten. Über seine MIDI OUT-Buchse kann es mit Expandern, Sequenzern, Synthesizern und PCs verbunden werden. Der PC-200 besitzt keine eigene Tonerzeugung.

11.5 Steinberg Cubasis

Steinberg Cubasis ist ein Midi-Sequenzer mit integriertem Noteneditor. Die über das MIDI-Interface am PC eintreffenden Midi-Daten können in Realzeit verarbeitet und unter Berücksichtigung des zeitlichen Aspekts in einer MIDI-Datei abgespeichert werden. Zum Aufnehmen stehen 16 Spuren zur Verfügung. Ungenauigkeiten im Timing können durch Quantisierung korrigiert werden. Die Darstellung eines Musikstücks erfolgt wahlweise in konventioneller Notenschrift, als *Piano-Rolle* oder als Liste von MIDI-Events. In allen Editoren können zahlreiche musikalische Kenngrößen wie Tonhöhe, -Lautstärke, -länge, Anschlagstärke interaktiv manipuliert werden. Der Noteneditor erlaubt die Wahl der Notenschlüssel, der Tonart, der Taktart und weiterer Darstellungsparameter. Das aufbereitete Notenbild läßt sich über die Druckerschnittstelle zu Papier bringen.



Screenshot vom Midisequenzer Steinberg Cubasis

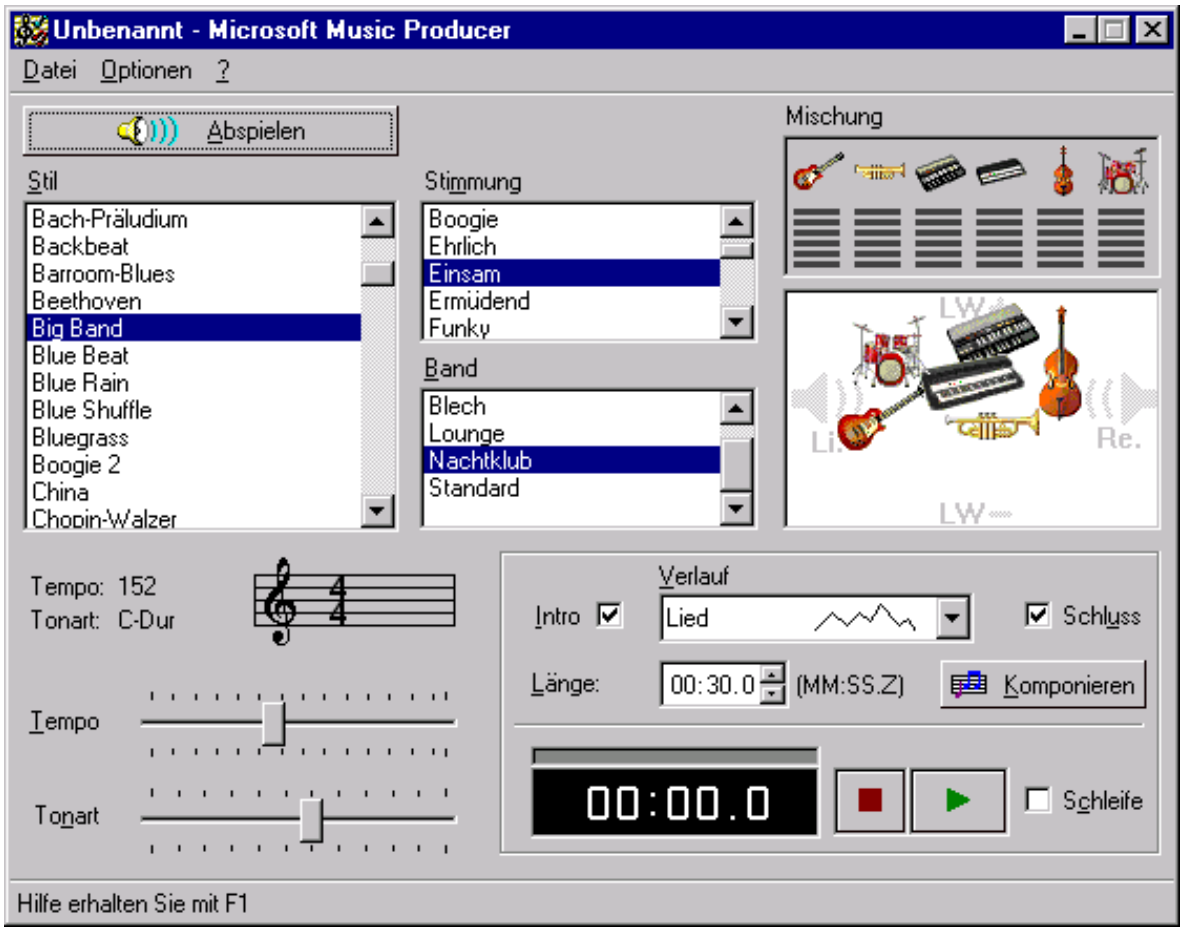


Notensatz, generiert von Steinberg Cubasis

Screenshot vom Midi-Plugin

11.6 Microsoft Music Producer

Der Microsoft Music Producer komponiert nach Vorgabe von Instrumentierung, Tempo, Tonart, Stil und Stimmung ein Musikstück bestimmter Zeitdauer, entweder als Schleife oder mit Intro und Schluß. Die generierte Datei kann im Midiformat exportiert und ohne Urheberrechts- oder Nutzungsgebühren auf einer Web-Seite verwendet werden.



Screenshot von Microsoft Music Producer

Kapitel 12

Video Analog

Der Begriff *Video* stammt von dem lateinischen Begriff *videre* (sehen) und bezeichnet Sequenzen von bewegten Bildern, die zur Bildschirmausgabe geeignet sind.

Als Aufnahmegerät verwendet man seit den 70er Jahren typischerweise CCD-Kameras (*Charge Coupled Device*). Hinter einem Linsensystem sind Tausende von Speicherzellen homogen auf einer Chip-Ebene verteilt. Jede Speicherzelle besteht aus einem Kondensator und einer Photodiode, über die der Kondensator bei Lichteinfall aufgeladen wird. Das Auslesen der Zellen geschieht zyklisch. Consumer-Camcorder verwenden 1/4-Zoll-Chips mit $576 \cdot 768 \approx 440.000$ Bildelementen; herstellbar sind CCD-Chips mit $2048 \times 2048 = 4.2$ Millionen Zellen à 9 μm Kantenlänge auf einer Fläche von der Größe einer Briefmarke. Durch Verschieben der CCD-Zellen um einige Nanometer lassen sich lichtempfindliche Zellen zwischen den CCD-Zellen simulieren. Als Stellglieder dienen *Piezo-Quarze*. Hierdurch entsteht eine Auflösung mit Dia-Qualität, allerdings werden für eine Aufnahme mehrere Sekunden benötigt. Zur Bildwiedergabe wird ein luftleer gepumpter Glaskolben (Bildröhre) verwendet, in dessen Inneren ein durch Magnetspulen abgelenkter Elektronenstrahl auf eine Phosphorschicht trifft und dadurch für eine Lichtemission sorgt.

Im Gegensatz zu den Druckwellen eines akustischen Signals kann eine diskrete Folge von Einzelbildern als kontinuierliche Sequenz wahrgenommen werden. Die Grenzfrequenz liegt bei etwa 16 Hz. Allerdings entsteht bis etwa 50 Hz ein Flimmereffekt durch die unvollkommene Speicherwirkung des Auges für optische Reize. Bei Filmprojektoren wird während der Projektion eines Bildes der Lichtstrom zusätzlich zweimal unterbrochen, und man erreicht dadurch eine Bildauffrischungsfrequenz von $3 \times 16 = 48$ Hz. Bei Fernsehgeräten wird ein Vollbild in zwei zeilenweise ineinandergeschachtelte Halbbilder geteilt. Es wird jeweils ein Halbbild nach dem anderen im Zeilensprungverfahren übertragen (*Interlace*-Verfahren). Jedes Halbbild wird 25mal pro Sekunde dargestellt, also beträgt die Vertikalfrequenz eines Vollbildes 50 Hz, die Zeit zwischen zwei Halbbildern 20 msec.

12.1 Schwarz-Weiß-Fernsehen

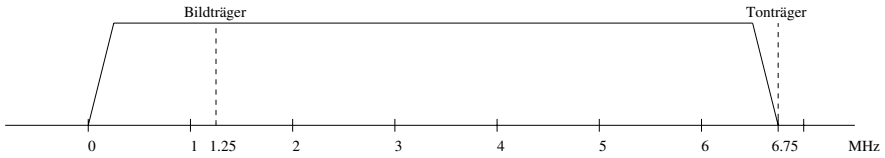
Ein Fernsehbild enthält nach der in Europa gültigen Norm 625 Zeilen (davon 575 sichtbar) und hat ein Verhältnis Breite zu Höhe von 4 : 3. Daraus resultieren $625 \cdot \frac{4}{3} = 833$ Bildpunkte pro Zeile. Hierfür werden 833/2 Schwingungen benötigt. Bei 625 Zeilen und 25 Hz ergibt sich

eine Grenzfrequenz von 6.5 MHz. Verwendet werden in der Praxis aber nur 5 MHz, was 320 Helligkeitsschwankungen = 640 Bildpunkten entspricht. Jedes Bild wird Zeile für Zeile von links nach rechts geschrieben. Am Zeilenende springt der Strahl dunkel nach vorne zurück (horizontale Austastlücke). Für jede Zeile steht zur Verfügung

$$\frac{1s}{25 \cdot 625} = 64\mu s$$

Am Bildende springt der Strahl dunkel nach oben zurück (vertikale Austastlücke).

Zur Übertragung der Bild-, Austast-, Synchroninformation (BAS-Signal) setzt man auf einen Bildträger von 1.25 MHz die 5.5 MHz Videobandbreite als Restseitenbandmodulation drauf. Der Ton benutzt ein frequenzmoduliertes Signal mit einem Tonträger, der 5.5 MHz über dem Bildträger liegt.



Spektrum eines BAS-Signals

12.2 Farbfernsehen

Zusätzlich zu den Grauwerten müssen die Farbinformationen übertragen werden. Eine geeignete Zerlegung eines RGB-Signals bietet das YUV-Modell, welches den drei Farbwerten Rot, Grün, Blau einen Luminanzwert und zwei Farbdifferenzwerte zuordnet.

$$Y = 0.30R + 0.59G + 0.11B$$
$$U = (B - Y) \cdot 0.493$$
$$V = (R - Y) \cdot 0.877$$

Für die technische Aufbereitung zu einem einzigen Farb-, Bild-, Austast-, Synchronsignal (FBAS, *Composite Signal*) wurden verschiedene Normen geschaffen:

NTSC *National Television Systems Committee*
verbreitet in den USA
525 Zeilen, 30 Hz
Quadraturamplitudenmodulation mit Farbträger bei 4.429 MHz.

SECAM *Sequential Coleur avec Memoire*
verbreitet in Frankreich und Osteuropa
625 Zeilen, 25 Hz
Frequenzmodulation

PAL *Phase Alternating Line*
verbreitet in Westeuropa außer Frankreich
625 Zeilen, 25 Hz
Quadraturamplitudenmodulation mit Farbträger 4.4336 MHz neben dem Bildträger: Farbträger wird mit Farbdifferenzsignal U multipliziert. Der um $90\,^\circ$ verschobene Farbträger wird mit Farbdifferenzsignal V multipliziert. Beide Ergebnisse werden addiert.

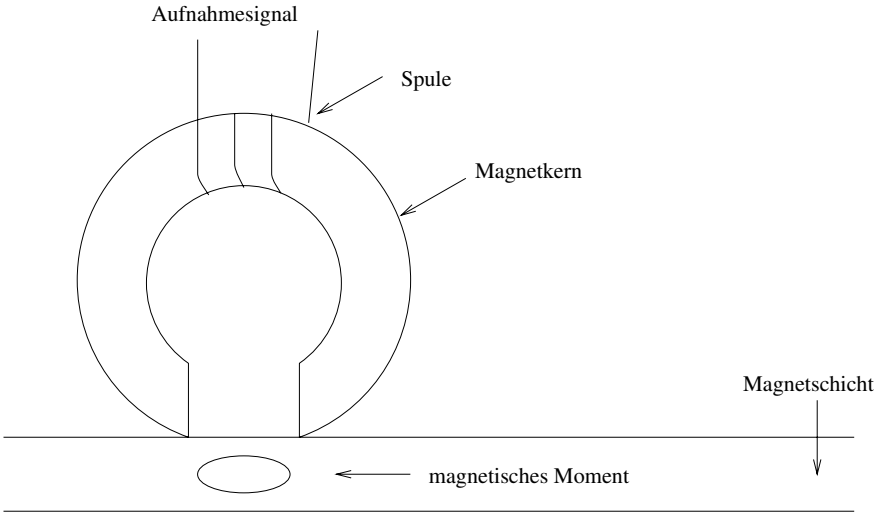
D2-MAC *Duobinary Multiplexed Analogue Components*
Als PAL-Plus in Deutschland eingeführt.
Zwischenstufe vom heutigen Fernsehen zu HDTV
Zeitmultiplexverfahren mit digitalem Ton, eine Zeile $= 64\mu s$ teilt sich auf in
19 % für 105 Bit Ton und Daten
27 % für Chrominanz $U + V$
54 % für Luminanz Y
Audio wahlweise $2\times$ Hifi Stereo oder $8\times$ Mono
105 Bit in $64\mu s$ ergibt 1.64MBit/s.

HDTV *High Definition Television* auch genannt
HD-MAC *High Definition Multiplexed Analogue Components*
Breite zu Höhe = 16 : 9
1250 Zeilen, $\sim$ 2000 Pixel pro Zeile
Höhere Bildschärfe durch Luminanzbandbreite 30 MHz
50 Halbbilder pro Sekunde
Bildwiederholfrequenz im Empfänger 100 Hz
Ton in Stereo-CD-Qualität
Gegenüber PAL werden also doppelt soviel Zeilen in der Hälfte der Zeit gesendet. Somit steigt die Datenrate um den Faktor

$$2 \cdot 2 \cdot \frac{16/9}{4/3} = 5.33$$

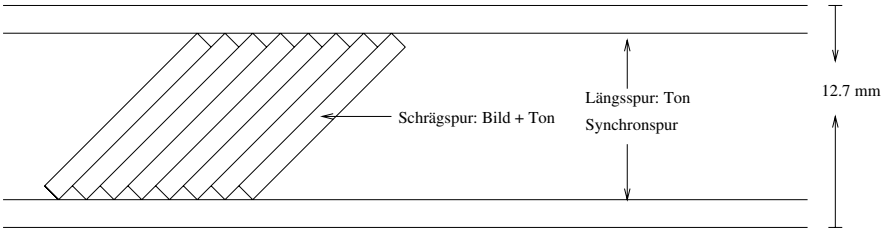
12.3 Videoaufzeichnung

Da bei der PAL-Fernsehnorm von den 625 Zeilen nur 575 sichtbar sind, entstehen bei einem Breiten/Höhenverhältnis von $4:3$ insgesamt 768×576 Bildpunkte. Ein Vollbild besteht dabei aus 2 Halbbildern mit je 288 Zeilen.



Prinzip der magnetischen Speicherung eines Signals

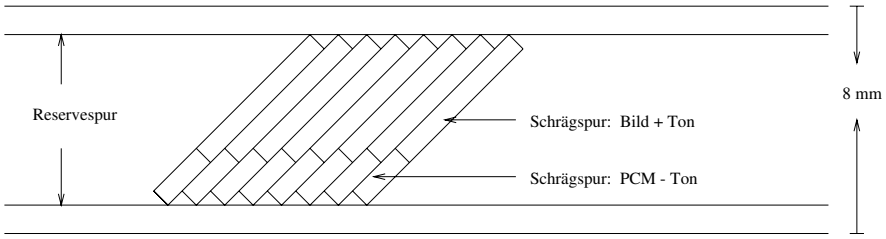
Beim *Video Home System* (VHS) wird ein 1/2-Zoll-Magnetband verwendet, auf dem mit rotierenden Videoköpfen die Bild- und Toninformation in Schrägspuren und mit einem feststehenden Tonkopf der Ton zusätzlich auf der Randspur abgelegt wird.



VHS-Aufzeichnung

Da das Band mit 2.4 cm/sec angetrieben wird, erreicht der Ton auf der Randspur keine Hifi-Qualität. Durch die rotierenden Videoköpfe beträgt die relative Bandgeschwindigkeit 20 cm/sec, wodurch ein Hifi-Stereo-Ton ermöglicht wird, der aber unwiderruflich mit dem Bild vermischt ist. Das Aufbringen eines neuen Hifi-Tons ist nur beim Kopieren möglich. Einige Hifi-VHS-Recorder können den Originalton von der Schrägspur auf die Randspur lippensynchron unter Zumischung weiterer Tonquellen überspielen (Audio-Dub). VHS-C verwendet dasselbe Aufnahmeformat mit verkleinerten Kassettenabmessungen. Dadurch reduziert sich die Laufzeit auf 30 bis 45 Minuten.

1985 entstand durch den Zusammenschluß von 120 Firmen das spezielle Filmer-System *Video8*. Verwendet wird ein 8 mm Band in einer Konfektionierung von 30-, 60- oder 90-Minuten-Kassetten. Bei einigen Camcordern wird in einem separaten Abschnitt der Schrägspur der Ton im 8-Bit-PCM-Format abgelegt und kann dort unabhängig vom Bild manipuliert werden. Es gibt allerdings keine Video-8-Recorder, die unter Beibehaltung des Tons eine neue Bildsequenz aufspielen.



Video-8-Aufzeichnung

Super VHS und Hi 8 sind Weiterentwicklungen von VHS bzw. Video 8 unter Beibehaltung der Kassettenformate. In beiden Fällen wurde die Zahl der horizontalen Linien von 288 auf 400 gesteigert. Außerdem werden Luminanz- und Chrominanz-Informationen separat aufgezeichnet und übertragen. Statt eines 2-poligen Cinch-Steckers bei VHS wird nun ein 4-poliger Hosiden-Anschluß benötigt.

Seit 1995 gibt es von Sony einen digitalen Camcorder. Wichtigste Merkmale sind 500 horizontale Linien und 16-Bit-PCM-Ton.

12.4 Time Code

Zusätzlich zur Bild- und Toninformation wird auf dem Magnetband ein fortlaufender Zeitstempel (Stunde/Minute/Sekunde/Frame) abgelegt, damit ein Schnittsteuergerät einzelne Szenen bildgenau abrufen kann.

LTC *Longitudinal Time Code*
abgelegt in Audio-Längsspur
lesbar nur während Wiedergabe
geht beim Kopieren teilweise verloren.

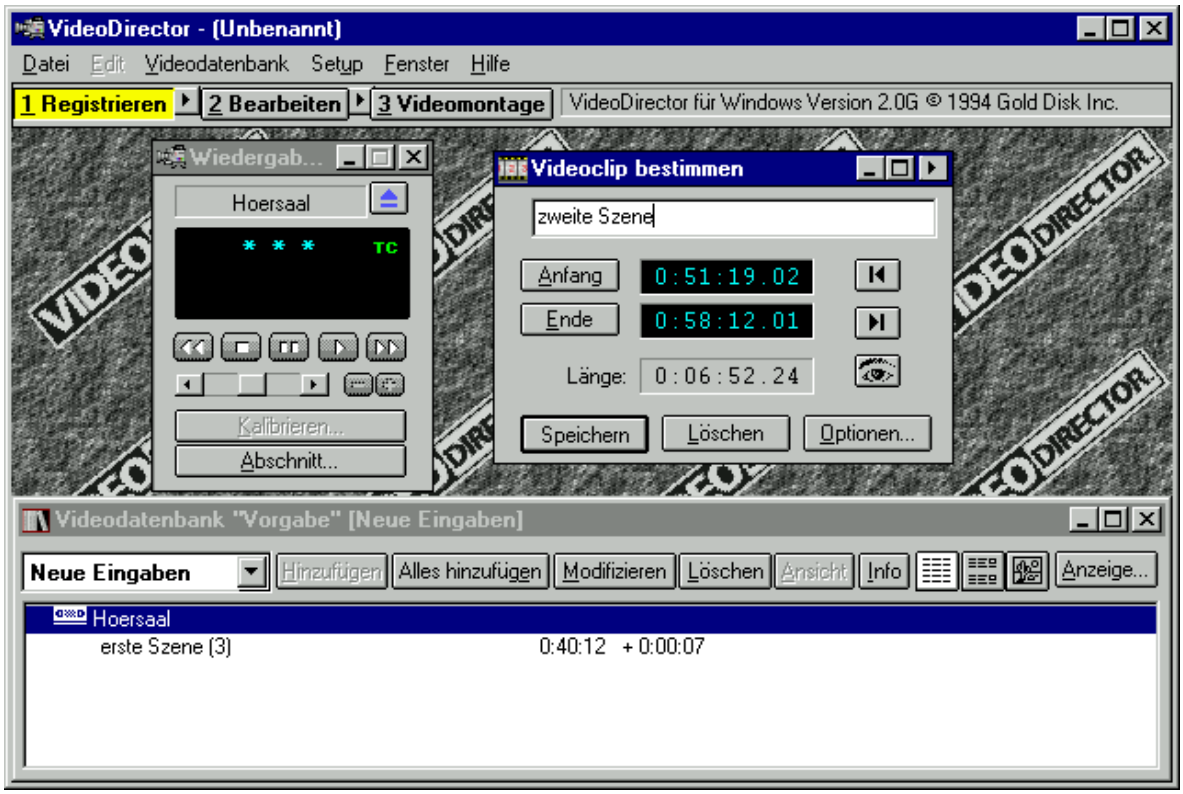
VITC *Vertical Interval Time Code*
abgelegt in vertikaler Austastlücke.
Problem: dort liegt auch Videotext
lesbar während Wiedergabe und Suchlauf
nur mit Bild gleichzeitig ablegbar.

RAPID abgelegt in VHS-Synchronspur
lesbar während Wiedergabe und Vor-/Rücklauf
nachträglich ablegbar.
Problem: wegen Längsspur nur 1 Code pro 2 sec.

RCTC *Rewritable Consumer Time Code* (Sony)
abgelegt auf Video-8-Schrägspur
zwischen Bild und PCM-Abschnitt
lesbar während Wiedergabe und Vor-/Rücklauf
auch nachträglich ablegbar.

12.5 Gold Disk Video Director

Gold Disk Video Director ist ein Werkzeug für den linearen Videoschnitt. Das Quellmaterial befindet sich in analoger Form auf der Kassette eines Camcorders. Der Camcorder muß über eine LANC-Buchse verfügen, die Signale zur Laufwerkssteuerung entgegennehmen kann. Der aufnehmende Videorecorder muß über eine Infrarot-Fernbedienung verfügen. An einem COM-Port des PCs wird ein Kabel angeschlossen mit einem LANC-Stecker für den Zuspieler und ein Kabel mit einem Infrarotsender für den aufnehmenden Videorecorder. Das Videosignal fließt vom Camcorder direkt zum Videorecorder und wird über einen dort angeschlossenen Monitor angezeigt. Im ersten Arbeitsgang wird das Quellmaterial linear gesichtet und interaktiv durch Setzen von Anfangs- und Endemarken eine Liste von Videoclips erstellt. Im zweiten Arbeitsgang werden die Clips durch Vertauschen in die gewünschte Reihenfolge gebracht. Diese Reihenfolge wird im dritten Arbeitsgang per Programm abgearbeitet, wenn der Camcorder über seine LANC-Buchse angesteuert wird, um jeweils den nächsten Clip auf den Videorecorder zu überspielen. Beim Aneinandersetzen kann eine Genauigkeit von etwa ±3 Einzelbildern erreicht werden, wenn der Zuspieler über einen Time-Code verfügt.



Screenshot von Videoschnittwerkzeug Golddisk VideoDirector

Kapitel 13

Video Digital

13.1 Videoformate

Apple Quicktime ist ein Dateiformat zum Speichern und Synchronisieren von Video- und Audioinformationen. Ursprünglich nur für Macintosh-Plattformen entwickelt, ist es inzwischen auch unter Windows und UNIX abspielbar. Auf der PC-Plattform ist als Grundlage für Video for Windows das AVI-Format (Audio/Video-Interleaved) sehr verbreitet.

Screenshot vom Quicktime-Plugin

13.2 MPEG

25 PAL-Fernseh-Bilder pro Sekunde im Format 768 × 576 bei 24 Bit Farbtiefe ergeben einen Datenstrom von 32 MBytes/sec. Auf eine CD mit 660 MByte Speicherkapazität würde demnach ein 20-Sekunden-Film passen. Aber zum Abspielen durch ein Single-Speed-CDROM-Laufwerk wäre die übliche Transferrate von 1.34 MBit/sec um den Faktor 180 zu niedrig. Drei Komponenten tragen zur Datenreduktion bei:

1. Vereinfachung des Video-Signals (*Subsampling*, Faktor 4)
2. Ausnutzung räumlicher Redundanz (JPEG, Faktor 15)
3. Ausnutzung zeitlicher Redundanz (MPEG, Faktor 3).

Da bei sich bewegenden Motiven zwei Halbbilder mit einer zeitlichen Verzögerung von 1/50 sec = 20 msec entstanden sind, können sie nicht exakt zu einem konsistenten Vollbild vereinigt werden. Daher stützt sich die Kompression eines Einzelbilds nur auf ein Halbbild mit halbierter Auflösung. Von 575 sichtbaren PAL-Zeilen bleiben also nur 288 übrig. Entsprechend wird auch die horizontale Auflösung in etwa halbiert, und zwar auf 352 Pixel pro Zeile. Als Source-Input-Format (SIF) für die Digitalisierung wurde daher festgelegt

Europa:	352 ×288 à 25 Hz
USA:	352 ×240 à 30 Hz.

In beiden Fällen beträgt bei 24 Bit Farbtiefe die Datenrate 58 MBit/sec. Das SIF wird durch ein Preprocessing vor der Digitalisierung bereitgestellt.
Zunächst wird die Farbinformation in das YUV-Modell überführt und ein 4 : 1 : 1 Subsampling angewendet. D.h. unter Beibehaltung der Helligkeitsauflösung werden beide Farbdifferenzauflösungen halbiert. Somit entstehen pro PAL-Bild

1	Luminanzmatrix	352 × 288
2	Crominanzmatrizen	176 × 144

mit insgesamt $22 \cdot 18 = 396$ Makroblöcken, bestehend aus 16 × 16 Blöcken der Luminanzmatrix und zwei assoziierten 8 × 8 Blöcken der Chrominanzmatrizen.

Eine Videosequenz wird in Gruppen unterteilt. Alle Gruppen haben die gleiche Anzahl von Bildern. Die Gruppen werden zusammenhängend komprimiert und erlauben den unmittelbaren Zugriff nur auf das Anfangsbild. Da drei Zugriffsmöglichkeiten pro Sekunde möglich sein sollen, enthält eine Gruppe je nach Videonorm 8-10 Bilder.

I-Picture (*Intro Coded Picture*)
: Das Anfangsbild einer Gruppe wird auf der Basis von 8 × 8 Blöcken einer JPEG-Kompression unterworfen, d.h. Diskrete Cosinus Transformation, Quantisierung, Differenzkodierung der DC-Koeffizienten und Lauflängenkodierung der AC-Koeffizienten mit Huffman-Tabellen.

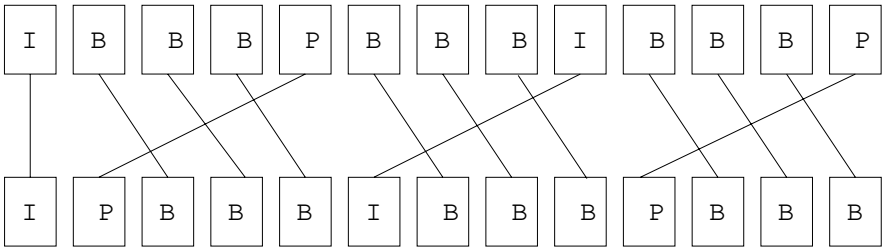
P-Picture (*Predictive Coded Picture*)
: Ein *P-Picture* wird mit Bezug auf ein Referenzbild kodiert, welches durch ein voran-

gegangenes *I-* oder *P-Picture* gegeben ist. Hierbei wird ausgenutzt, daß sich in zeitlich aufeinanderfolgenden Einzelbildern gewisse Bildbereiche komplett verschieben, z.B. durch einen Kameraschwenk oder durch ein sich bewegendes Objekt. Es wird daher zu jedem Macro-Block des *P-Pictures* ein möglichst ähnlicher Macroblock im Referenzbild gesucht. Gespeichert wird der Verschiebungsvektor und ggf. eine Differenzmatrix mit den beobachteten Pixelabweichungen. Für die Folge der Bewegungsvektoren wird eine DPCM-Kodierung verwendet. Läßt sich kein geeigneter Macroblock finden, wird die für *I-Picture* vorgesehene Kompression angewendet.

B-Picture (*Bidirectionally predictive Coded Picture*)
Ein *B-Picture* bezieht sich auf ein vorangegangenes *P-* oder *I-Picture*, auf ein nachfolgendes *P-* oder *I-Picture* oder auf beide. In diesem Fall wird die Differenz zur Interpolation zweier Macroblöcke abgespeichert. Dies ist sinnvoll bei sich bewegenden Objekten, die sukzessive einen Teil des bislang verdeckten Hintergrunds sichtbar werden lassen.

D-Picture (*DC-Coded Picture*)
D-Pictures enthalten *I-Picture*-Kodierungen eingeschränkt auf die DC-Koeffizienten. Hierdurch wird eine eingeschränkte Bildqualität bei schnellem Vor- oder Rücklauf bereitgestellt.

Zur Kompression und Dekompression muß die ursprüngliche Bildfolge umgestellt werden.



Umstellung der Bildfolge

Während der Anzeige des ersten *I*-Bildes wird das nachfolgende *P*-Bild decodiert. Die nächsten *B*-Bilder werden unmittelbar nach ihrer Decodierung angezeigt. Nun kann das bereits decodierte *P*-Bild angezeigt werden, während das nächste *I*-Bild decodiert wird.

Zur Synchronisation wird spätestens nach 16 Bildern eine Zeitmarke eingefügt und ggf. mit einem Audiostrom verschachtelt. Bei der Wiedergabe kann der Dekodierer aus den Zeitmarken ersehen, ob er bei einem Synchronisationsverlust Bilder überspringen oder wiederholen muß.

Da Videorecorder ihre Bildsequenzen mit 25 Hz abliefern, muß die Kodierung in Realzeit erfolgen. Zur Erzeugung einer MPEG-Datei wird oft zunächst mit Hilfe einer Hardwarekompression eine Folge von *I-Pictures* erzeugt (Motion JPEG). Daraus entsteht dann offline per Software das MPEG-Format.

Screenshot vom MPEG-Plugin

13.3 H.261

1984 hat die CCITT eine Arbeitsgruppe gegründet, die eine Videokomprimierung mit einer Transferrate von 64 kBit/sec spezifizieren sollte. Hierdurch werden Videokonferenzen auf der Grundlage von ISDN-Verbindungen möglich. Das Verfahren beruht auf dem MPEG-Prinzip und erreicht eine weitere Kompression durch Weglassen von Bildern. Eine häufige Verwendung dieses Formats findet im MBone statt, ein System zum Broadcasten von Videokonferenzen im Internet über einen Spannbaum von Kommunikationsleitungen an eine Menge von angemeldeten Benutzern.

Screenshot vom Videokonferenzwerkzeug VIC

13.4 MPEG-2

Bei einer Vorgabe von 1.5 MBit/sec mußte als Digitalisierungsinput das Source-Input-Format von 352 × 288 Pixeln verwendet werden. Die Auflösung liegt damit unterhalb von VHS. Für eine verbesserte Qualität soll der Standard MPEG-2 sorgen. Hier wird eine Datenrate von bis zu 6 MBit/sec akzeptiert. Dadurch können Auflösungen von 1250 Zeilen mit einem Seitenverhältnis 16 : 9 verarbeitet werden. Außerdem ist die Integration mehrerer Formate in einem Videostrom vorgesehen, so daß sich die Wiedergabequalität vom Dekodierer skalieren läßt.

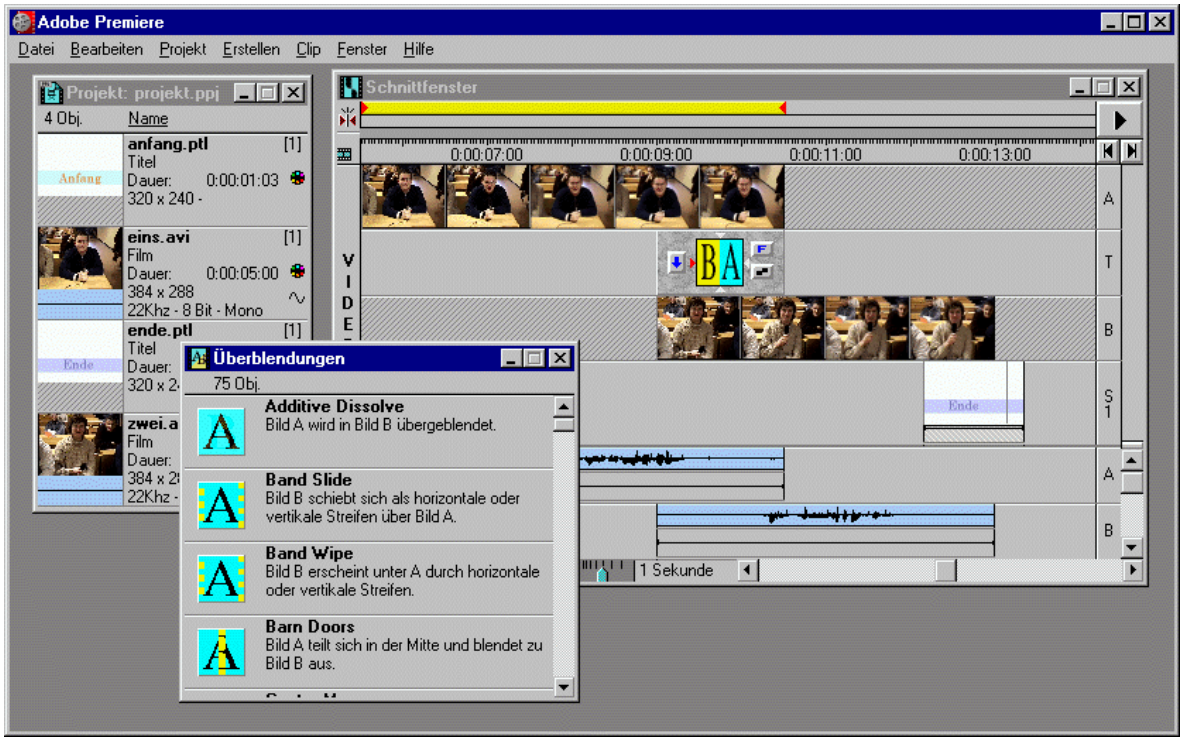
13.5 Fast FPS 60

Fast FPS 60 ist eine AT-Bus-Karte zur Kompression und Dekompression von Live-Video auf bzw. von der Festplatte in FBAS- und S-Video-Qualität. Durch die integrierte Overlay-Funktion kann das Videobild in einem Fenster beliebiger Größe in Echtzeit und True Color dargestellt werden. Die aus den Videosequenzen erzeugten Dateien verwenden das MJPEG-Format: eine Folge von JPEG-komprimierten Halbbildern. Bei bestmöglicher Kompressionsqualität (Faktor 13) benötigt eine Minute Video etwa 35 MByte Plattenplatz.

Screenshot vom Video-Capture-Werkzeug Fast FPS 60

13.6 Adobe Premiere

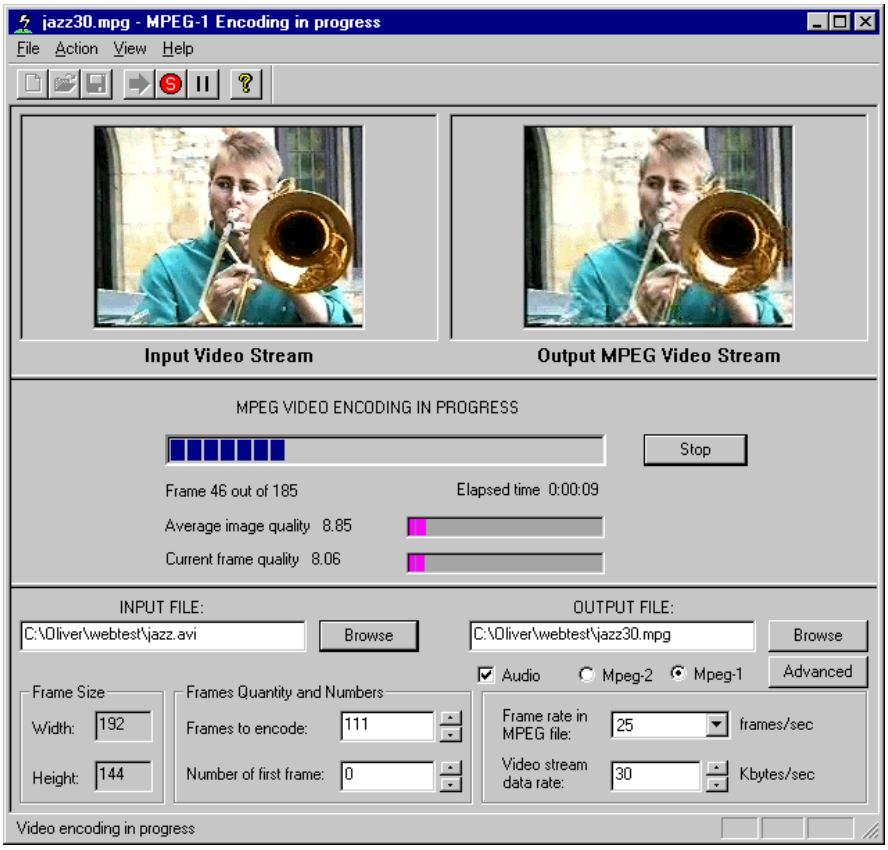
Adobe Premiere ist ein Werkzeug für den nichtlinearen Videoschnitt. Das Quellmaterial befindet sich in digitalisierter Form auf Festplatte. Durch Drag-and-Drop-Technik werden auf mehreren parallelen Zeitachsen Video- und Audio-Clips plaziert und mit Überblendungseffekten versehen. Bei der endgültigen Produktion werden alle beteiligten Sequenzen gemäß ihrer Lage im Zeitfenster zu einer AVI-Datei zusammengemischt. Die Erzeugung eines 60-Sekunden-Clips dauert etwa 20 Minuten.



Screenshot vom Videoschneidewerkzeug Adobe Premiere

13.7 Ligos LSX-MPEG-Encoder

Der Ligos LSX-MPEG-Encoder erzeugt per Software aus einer im AVI-Format vorliegenden Videodatei eine MPEG-1-Datei. Zur Konvertierung eines 60-Sekunden-Clips werden etwa fünf Minuten benötigt. Der Platzbedarf der Videodatei schrumpft etwa auf ein Drittel bei einer Qualitätsvorgabe ohne deutliche Artefakte.



Screenshot vom Ligos LSX-MPEG-Encoder

13.8 Video-on-Demand

Hoch komprimierte Videos verschiedener Formate sind zum Runterladen im Internet verfügbar. Die Tagesschau-Redaktion in Hamburg stellte bis Juli 1998 Videoclips im VDO-Format bereit.

Screenshot vom VDO-Plugin

Die amerikanische Nachrichtenagentur FoxNews verwendet das RealVideo-Format, das seit August 1998 auch von der Tagesschau verwendet wird.

Screenshot vom RealVideo-Plugin

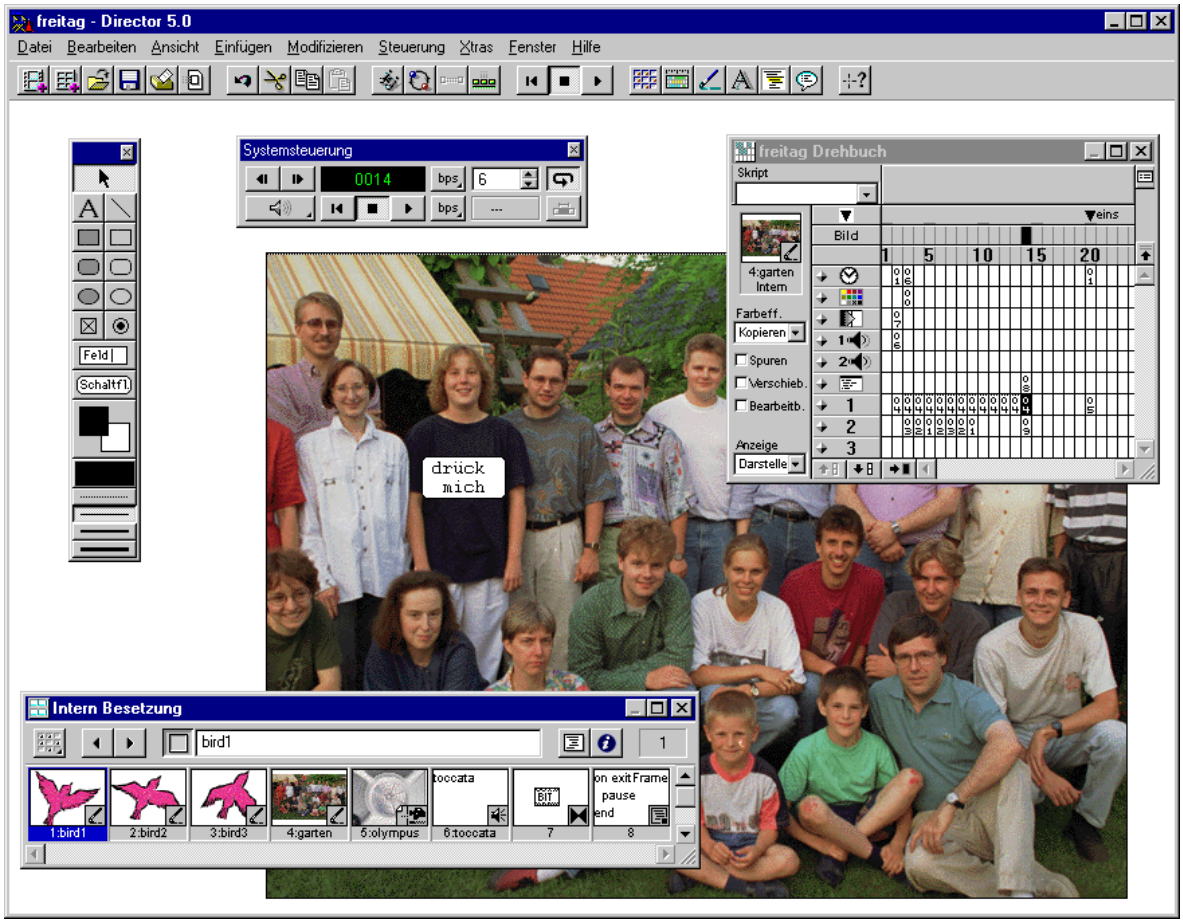
Kapitel 14

Autorensystem

Unter einem Autorensystem versteht man ein Werkzeug zur Unterstützung des Herstellungsprozesses einer Multimedia-Applikation. Integrale Bestandteile sind Hilfsmittel zur Medienbearbeitung und zum Screendesign sowie ein Strukturplaneditor. Ein Strukturplaneditor, der die räumliche und zeitliche Koordination aller beteiligten Medien ermöglicht.

14.1 Macro Media Director

Macro Media Director ist ein Autorensystem mit einem Matrix-orientierten Strukturplaneditor: Eine Spalte entspricht einem Zeitintervall mit parametrisierter Dauer; eine Zeile repräsentiert einen Medienkanal, der mit den Darstellern aus der Besetzungsliste (Text-, Bild-, Audio-, Video-Dokumente) ggf. in jeweils modifizierter Form, gefüllt ist. Überblendungen zwischen den Spalten sind möglich. Ein spezieller Kanal enthält zusätzliche Anweisungen zum Programmablauf in Form von Lingo-Scripts.



Screenshot vom Strukturplaneditor Macromedia Director