Skriptum zur VL

MULTIMEDIA I

Mag.Dr. Andreas Uhl

RIST++ und Institut für Scientific Computing

Universität Salzburg

Adresse:

Andreas Uhl
Forschungsinstitut für Softwaretechnologie
Hellbrunnerstr.34
A-5020 Salzburg
Österreich
Tel.: ++43/(0)662/8044/5308
Fax: ++43/(0)662/8044/138
E-mail: uhl@cosy.sbg.ac.at

Inhaltsverzeichnis

Kapitel 1

Video

Im Zusammenhang mit der Verarbeitung, mit dem Übertragen und der Speicherung von Videodaten ist es selbstverständlich, dass diese Aufgaben effizient nur mit einem reduzierten Datenvolumen zu bewältigen sind. Das ISO MPEG Kommitee (Motion Picture Experts Group) ist das wichtigste Standardisierungsorgan in diesem Bereich. Zusätzlich zur Redundanz im Bildbereich die bei Stillbildkompression ausgenutzt wird kommt im Bereich der Videoverarbeitung noch die temporale Redundanz, d.h. die Ähnlichkeit aufeinanderfolgender Frames.

1.1 MPEG-1

MPEG-1 ist ein Standard für effiziente Speicherung und Übertragung von Video und Audio mit einer Datenrate von ca. 1.5 Megabit/sec. Klassisches Hauptanwendungsgebiet ist die Speicherung auf CD-ROM. Es wird nur progressive Video mit 8 bpp und einem Luminance und zwei Chromakanälen (Subsampling 4:2:0) berücksichtigt. Wesentlich ist, dass nur der Bitstream bzw. der Decoder standardisiert ist, der Encoder Teil muss nur den entsprechenden Bitstream generieren, ansonstern kann er frei designed werden. Ein Flag im Bitstream zeigt an ob es sich um "constrained parameter" MPEG-1 handelt, der von allen MPEG-2 Decodern verarbeitet werden kann. Im MPEG-1 Standard gibt es zwei grundsätzlich verschiedene Möglichkeiten, Videomaterial zu speichern.

1.1.1 Intraframe Kompression

In diesem Modus wird jeder Frame einer Videosequenz als unabhängiges Bild betrachtet und komprimiert. Als Konsequenz werden nur Redundanzen im Bildbereich ausgenutzt, die Kompression ist ähnlich wie bei MJPEG (= Motion JPEG). Obwohl die erreichbare Kompressionsrate nicht über die eines Stillbildkompressionsverfahrens hinausgeht, werden solche Verfahren dennoch für spezielle Anwendungen benötigt: z.B. für Video editing wo random access eine wesentliche Rolle spielt. Die Kompression ist dem JPEG Verfahren sehr ähnlich, mit einem wesentlichen Unterschied:

- Adaptive Quantisierung der AC Koeffizienten wird ermöglicht, kann von Makroblock zu Makroblock wechseln (ein Makroblock besteht in MPEG-1 aus 4 8x8 Pixel Blöcken Luminance und je einem 8x8

Chroma Block). Dies wird durchgeführt in Abhängigkeit von der Aktivität (der Unruhe) des Blocks und des Buffer Status bei Applikationen mit konstanter Bitrate.

Ansonsten werden in Analogie zu JPEG 8x8 Pixel Blöcke einer DCT unterzogen, die resultierenden AC Koeffizienten durch eine Quantisierungsmatrix quantisiert, zig-zag gescannt und mit gegebenen VLC Tabellen kodiert. Die DC Koeffizienten werden gleichmässig quantisiert und die verschiedenen DC Koeffizienten innerhalb eines Makroblocks und benachbarter Makroblöcke DPCM kodiert. Die sogenannten *I-Frames* bestehen ausschliesslich aus Makroblöcken die in der beschriebenen Art komprimiert wurden.

1.1.2 Interframe Kompression

In diesem Modus werden spatiale und temporale Redundanzen ausgenutzt und es wird versucht im Video auftretende Bewegungen (Objekt- oder Kamerabewegungen) durch vorausgehende Bildanalyse zu kompensieren. Die Bewegungsanalyse wird auf der Basis von Makroblöcken durchgeführt, d.h. für einen Makroblock gibt es einen sg. Displacementvektor, der angibt, wohin sich der entsprechende Makroblock bewegt hat. Das bedeutet, dass das erlaubte Bewegungsmodell auf Translation von Blöcken eingeschränkt ist. Die Art der Berechnung des Displacementvektors ist nicht standardisiert und ist spezifisch für die jeweilige Implementierung. Das klassische Verfahren zur Bewegungsschätzung (Motion estimation) ist sg. Blockmatching das in allen Hardwareimplementierungen verwendet wird. Bei diesem Verfahren wird ein Block im zu kodierenden Frame mit allen möglichen Blöcken in einem Suchfenster bestimmter Grösse im bereits *rekonstruierten* Frame verglichen und auf den ähnlichsten Block zeigt dann der Displacementvektor. (Es ist wichtig, dass ein bereits rekonstruierter Frame verwendet wird, denn der Decoder hat ja nur solche Frames zur Verfügung - daraus folgt, dass ein Encoder immer auch einen Decoder entrhalten muss). Sind für alle Blöcke im zu kodierenden Frame die entsprechenden Displacementvektoren gefunden, wird der "predicted" Frame aus den Blöcken des reference Frame berechnet (die "motion compensated prediction") und der Differenzframe zum zu kodierenden Frame berechnet. Gespeichert werden im MPEG-1 Bitstream dann die Displacementvektoren (wobei durch DPCM Kodierung weitere Redundanz zwischen benachbarten Vektoren - entsprechend grösseren sich bewegenden objekten - ausgenutzt wird) und die Differenzframes (welche wieder in 8x8 Pixelblöcke zerlegt einer DCT unterzogen werden. Die Quantisierungsmatrix weist an allen Stellen gleiche Werte auf, da keinesfalls hochfrequente Inhalte bei Errorframes unterdrückt werden dürfen, ausserdem wird der DC Koeffizient nicht gesondert behandelt.) Interessanterweise ist für die Motionvektoren Halbpixel-Genauigkeit erlaubt (die fehlenden Werte werden interpoliert, was insbesondere bei starken Bewegungen deutliche Qualitätsverbesserung bringt).

In Entsprechung zu den I-Frames gibt es zwei Frame-typen im interframe Kompressionsmodus:

- P-Frames: pro Makroblock gibt es maximal einen Displacementvektor der in die "Vergangenheit" zeigt. Je nach Qualität der erreichten Prediction werden Makroblöcke in P-Frames als Errorblöcke mit Displacementvektoren oder als I-Frame Blöcke gespeichert.
- B-Frames: pro Makroblock kann es bis zu zwei Displacementvektoren geben. Im Fall der bidirektionalen Prediktion werden als Reference Frames rekonstruierte I oder P Frames in der Zukunft und Vergangenheit verwendet, die Prediktion entsteht durch Mittelung. Ebenso können B-Frame Makroblöcke aber nur durch Prediction auf zukünftige oder vergangene Frames dargestellt werden, oder sie werden bei schlechter Prediktion als I-Frame Blöcke gespeichert.

Die in einem MPEG-1 Bitstream tatsächlich vorkommenden Muster von I, P, und B-Frames hängen ab von

- Anforderungen bezüglich Zugriff und Dekodierungsverzögerung
- Benötigter Kompressionsrate
- Inhalt des Videos

Eine GOP (group of Pictures) enhält zumindest ein I-Frame und beliebig viele P und B-Frames. Der maximale Abstand zwischen zwei I-Frames beträgt in MPEG-1 132 Frames !

1.2 MPEG-2, H.261 und H.263

1.2.1 MPEG-2

MPEG-2 zielt auf höhere Bitraten und damit höhere Qualität als MPEG-1. Erlaubt ist 4:2:2 und auch 4:4:4 Ausgangsmaterial. Tyische Anwendungen sind digitales Fernsehen (HDTV - High Definition TV) und Video auf DVD. Durch die Ausweitung auf den TV Bereich wird nun auch interlaced Video erlaubt (Interlaced Video besteht aus Paaren von sg. "Fields", die die halbe Zeilenauflösung eines gesamten Frames haben und zeitversetzt sind. Ursprünglich hatten sie die Aufgabe, flimmerfreies Fernsehen bei geringer Datenrate zu ermöglichen - in jedem Zeitschritt wurde nur jede zweite Zeile upgedatet). Daraus ergeben sich neue Möglichkeiten Bilder zusammenzusetzen: die sg. "Frame Pictures" entstehen durch Interleaving der even und odd Fields, "Field Pictures" sind nur halb so gross und sind entweder das even oder das odd Field. Diese beiden Typen können in einem MPEG-2 Bitstream beliebig abgewechselt werden. MPEG-2 hat eine Profile (Funktionalität) und Level (Parameterwerte) Struktur und könnte eher als Algorithmensammlung gesehen werden. Typische Schriebweise: Main Profile at Mail Level: MP@ML (z.B. DVD).

Zusätzliche Unterschiede zu MPEG-1 sind:

- Feinere Quantisierung der DC und AC Koeffizienten, unterschiedliche Quantisierung von Luma und Chroma
- Feinere Anpassung der adaptiven Quantisierung
- Displacementvektoren müssen mindestens mit Halbpixelgenauigkeit berechnet werden
- Durch das Interlacing gibt es viele zusätzlich erlaubte Arten Displacementvektoren zu berechnen wobei adaptiv zwischen Frame Picture basierter und Field Picture basierter Berechnungsart gewechslt werden kann (siehe auch dual-prime Prediction).
- Error Concealment: Es gibt Strategien wie durch Übertragungsfehler verlorengegangene Daten (Blöcke oder Displacementvektoren) interpoliert werden können, z.B. können für diesen Zweck auch für Intra-kodierte Blöcke Displacementvektoren gespeichert werden).
- Scalability (Skalierbarkeit): hier geht es darum aus ein und demselben Bitstream Videos unterschedlicher Auflösung, Qualität und Frameraten dekodieren zu können. In MPEG-2 wird das (im Vergleich zu Wavelet Methoden) sehr artifiziell durch sg. Base- und Enhancementlayer realisiert.

1.2.2 H.261

Ist der Videoteil des H.320 Standards, hauptsächlich vorgesehen für Videoconferencing und Bildtelephonie über ISDN Kanäle mit $p \times 64$ kbits/sec. H.261 ist MPEG/1 sehr ähnlich, erlaubt aber aus Komplexitätsgründen keine B-Frames (denn sowohl Encoder als auch Decoder müssen real-time fähig sein !). Ein H.261 Encoder kann auch Frames weglassen oder die Auflösung reduzieren um die Bildqualität bei gegebener Bitrate aufrechterhalten zu können. Zwei Bildformate werden unterstützt: Q(uater)CIF und CIF (common interchange format), chroma sampling ist 4:2:0.

1.2.3 H.263

Ist der Videoteil des H.324 Standards (Terminal for low-bit rate Multimedia Communications), hauptsächlich vorgesehen für Videoconferencing und Bildtelephonie über Telephonkanäle mit 28.8 kbits/sec.

Unterschide zu H.261:

- Zusätzliche Bildformate
- Displacementvektoren können mit Halbpixelgenauigkeit berechnet werden.
- Besseres Kodieren der Displacementvektoren.
- Optionales Arithmetisches Kodieren.
- Optionales Verwenden von überlappenden Blöcken bei der Berechnung der Displacementvektoren.
- Möglichkeit zur bidirektionalen Prediktion.

Weiter Verbesserungen in H.263+ wo u.a. ein grosser Bereich von Framegrössen erlaubt ist und das Intracoding durch Prediction von benachbarten Blöcken verbessert ist.

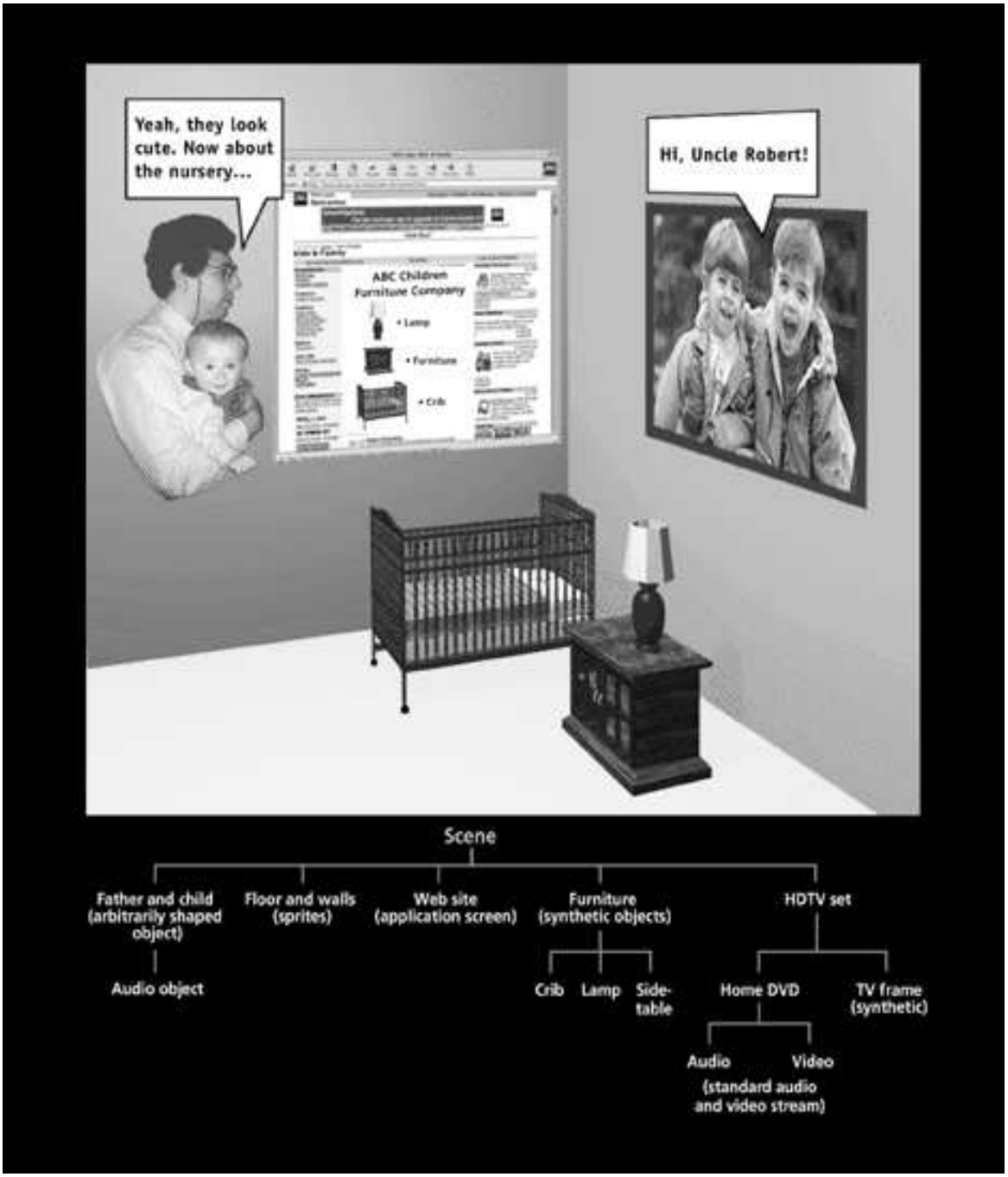
1.3 MPEG-4

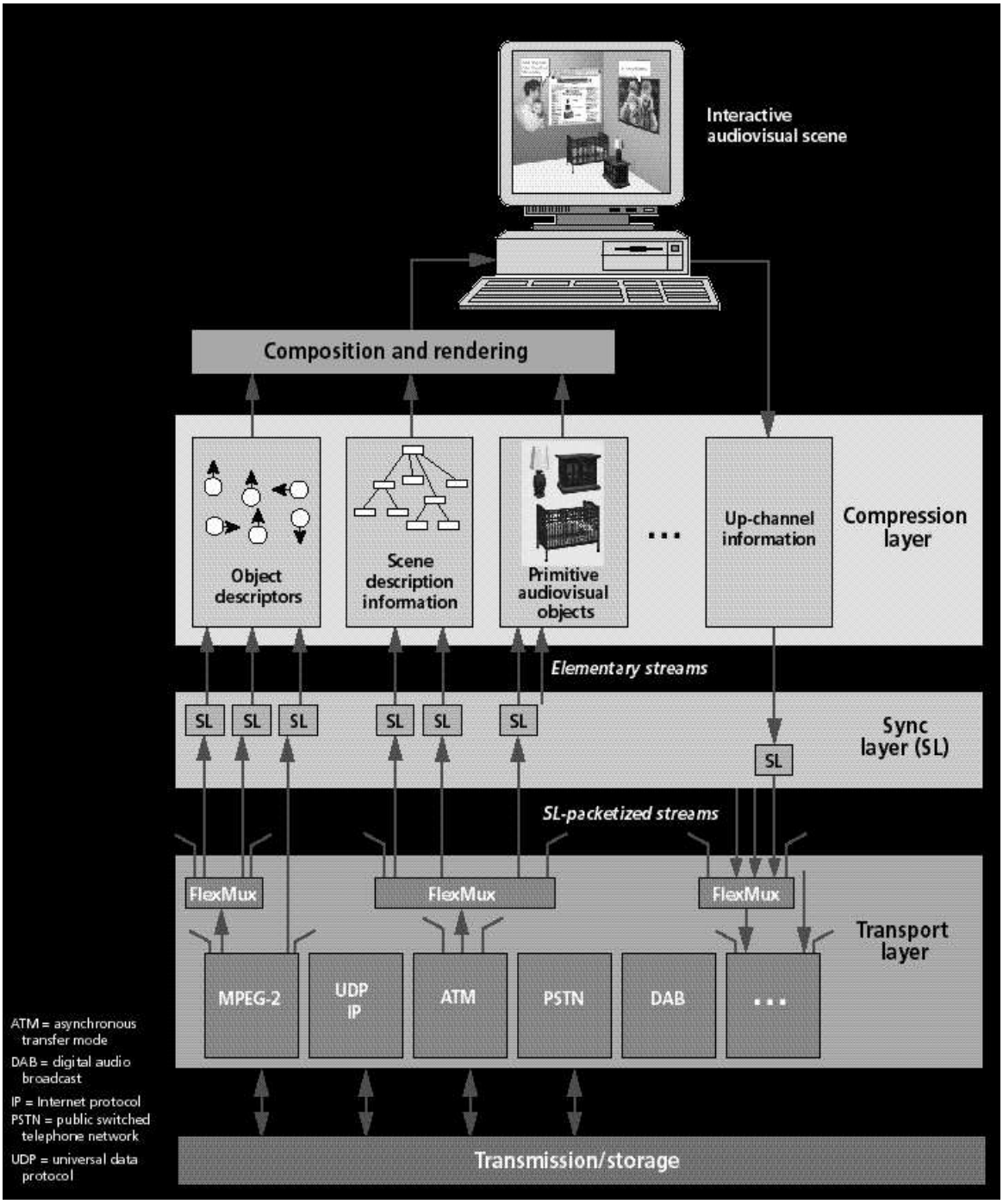
MPEG-4 ist ein Standard für Videokompression im Multimediabereich. Da ein wesentlicher Aspekt von vielen Multimedia-Anwendungen die Interaktivität ist, ist eines der Hauptzüge dieses Standards Methoden bereitzustellen, um Interaktivität zu ermöglichen. In bisherigen Standards ist Interaktivität durch die framebasierte Verarbeitung nicht möglich. In MPEG-4 wird das durch eine Objektbasierung ermöglicht.

MPEG-4 baut auf dem Erfolg in folgenden drei Feldern auf:

- Digitales Fernsehen
- Interaktive Graphik Anwendungen (z.B. Computerspiele)
- Interaktive Multimedia Anwendungen (z.B. WWW, Verteilung von/und Zugang zu Inhalt)

Dieser relativ neue Standard verbessert die Anwendungs/Verwendungsmöglichkeiten von Autoren, Service Providern und Verbrauchern.


Yeah, they look cute. Now about the nursery...
Hi, Uncle Robert!
ABC Children Furniture Company
Scene
Father and child (arbitrarily shaped object)
Audio object
Floor and walls (sprites)
Web site (application screen)
Furniture (synthetic objects)
Crib
Lamp
Side-table
HDTV set
Home DVD
Audio
Video
(standard audio and video stream)
TV frame (synthetic)

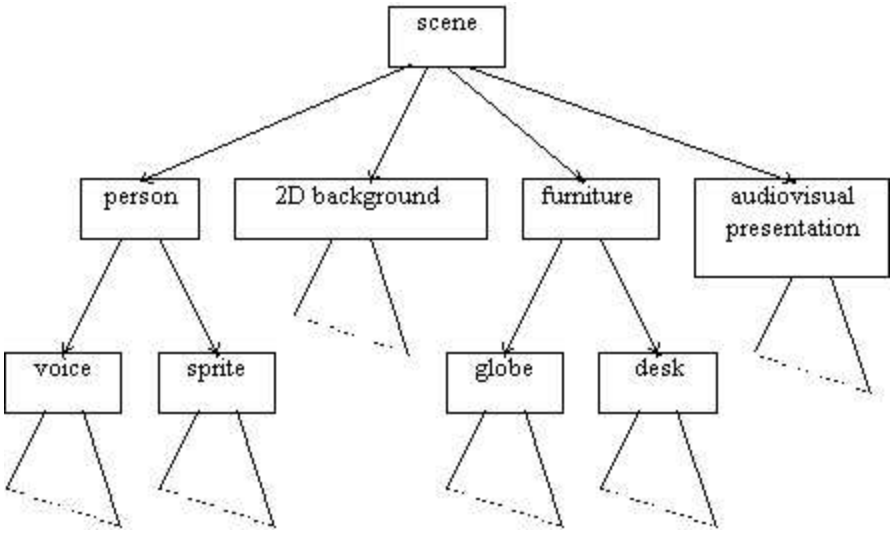

Interactive audiovisual scene
Composition and rendering
Object descriptors
Scene description information
Primitive audiovisual objects
...
Up-channel information
Compression layer
Elementary streams
SL
Sync layer (SL)
SL-packetized streams
FlexMux
Transport layer
MPEG-2
UDP IP
ATM
PSTN
DAB
...
ATM = asynchronous transfer mode
DAB = digital audio broadcast
IP = Internet protocol
PSTN = public switched telephone network
UDP = universal data protocol
Transmission/storage

- Autoren: Wiederverwertbarkeit von Information wird durch die Objektbasierung wesentlich verbessert, besserer Schutz von Copyright, etc.
- Service Provider: bessere Netzwerkauslastung durch MPEG-4 QoS Deskriptoren
- Verbraucher: erstmalige Möglichkeit von Interaktion mit Multimedialen Daten, von Standardnetzwerken bis hin zu mobilen Netzwerken (z.B. Interaktives Video Über Satellit auf personal organizer - die Frage bleibt: wer braucht das ;-)

MPEG-4 beinhaltet daher nicht nur (oder in erster Linie) Kompressionsverbesserung sondern Funktionalitätserhöhung. Das wird erreicht durch Standardisierung in folgenden Bereichen:

- Repräsentierung von "Media Objekten", die Einheiten von visuellen, audiovisuellen oder Audio Daten sind, die natürlichen oder künstlichen Ursprungs sein können.
- Zusammensetzung von media objekten zu einer audiovisuellen Szene.
- Multiplexing und Synchronisierung der Daten entsprechend den Media Objekten um sie über Netzwerke mit ihnen entsprechenden QoS Anforderungen transportieren zu können.
- Interaktion mit der audiovisuellen Szene beim Verbraucher.

MPEG-4 audioviduelle Szenen sind zusammengesetzt aus mehreren Media Objekten die in einer hierarchischen Art und Weise organisiert sind. Als Endknoten dieser Hierarchien findet man "primitive Mediaobjekte" wie Stillbilder, Video Objekte, Audio Objekte, u.s.w.

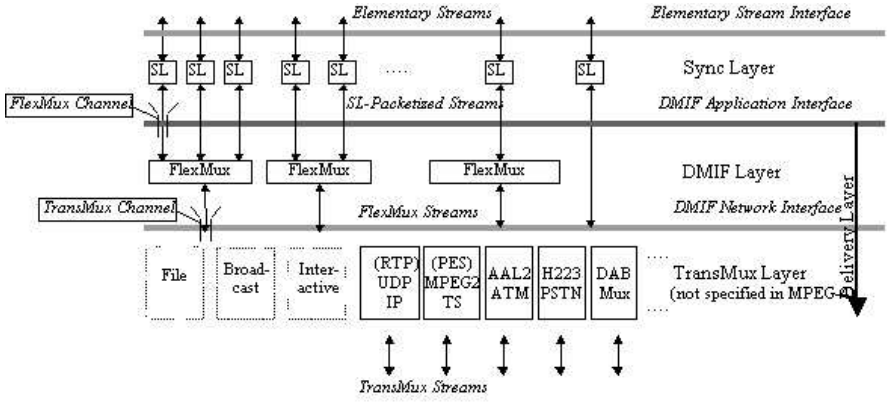


Zusätzlich gibt es noch Definition von speziellen Objekten, wie z.B. Text und Graphiken, sprechende synthetische Köpfe mit assoziiertem Text um die Sprache zu synthetisieren und den Kopf zu animieren, synthetischer Klang, etc.

Die Szenenbeschreibung ist standardisiert (mit einer grossen Nähe zur Computergraphik oder Virtual Reality - VRML) und erlaubt folgendes:

- Media Objekte können an beliebiger Plazierung in einem Koordinatensystem positioniert werden.
- Media Objekte können Transformationen unterzogen werden die die geometrische oder akustische Erscheinung verändern.
- Primitive Media Objekte können zu einem grossen media objekt zusammengefasst werden.
- Streamed Data kann auf Media Objekte angewendet werden um die Eigenschaften zu verändern (z.B. ein Klang, eine sich bewegende Textur, Animationsparameter für ein Gesicht, ...)
- Veränderung des Blick- und Hörpunkts des Benutzers in der Szene

Die synchronisierte Übertragung von streaming Data über ein Netzwerk mit verschiedenen QoS wird durch eine Synchonisierungsschicht und eine Übertragungsschicht mit einem zwei-Schicht Multiplexer geregelt.



Die erste Multiplexerschicht wird entsprechend der DMIF (Delivery Multimedia Integration Framework) Spezifikation implementiert. DMIF ist ein Sitzungsprotokoll für Multimedia Streaming Daten ähnlich wie FTP. Mit dem MPEG FlexMux Tool können z.B. elemetary streams mit ähnlichen QoS Anforderungen zusammengefasst werden. TransMux wird schliesslich verwendet um verschiedenen QoS Anforderungen gerecht werden zu können. Nur das Interface zu dieser Schicht ist standardisiert, die Umsetzung muss über das Netzwerk Transport Protokoll geschehen.

Möglichkeiten für Interaktion:

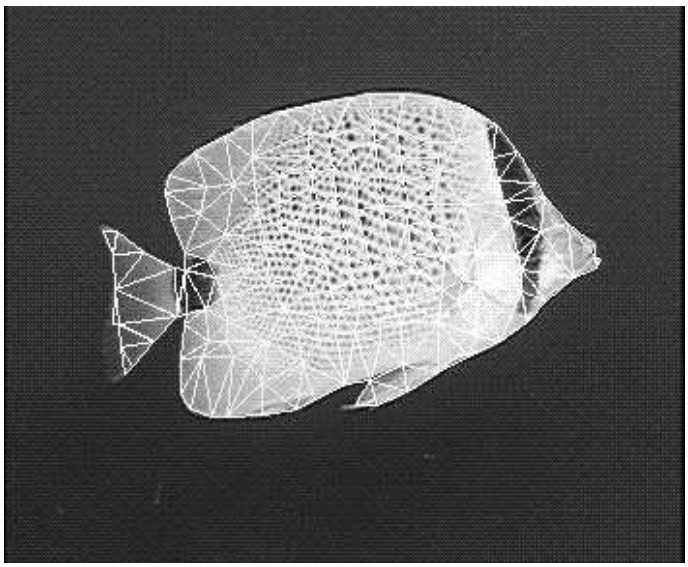
- Blick- und Hörwinkel verändern
- Objekte in einer Szene verschieben/entfernen
- Eine Kaskade von Ereignissen auslösen, durch “Mausklick” - Abspielen eines Videos oder animierter Graphik, ...
- Auswahl der geeigneten Sprache oder Ansicht bei Multiview/listen Ausnahmen

1.3.1 Kompression von synthetischen Videoobjekten

Exemplarisch werden hier Gisichtsanimation und mesh-coding mit Textur Mapping behandelt.

Ein "facial animation object" kann verwendet werden um ein bewegtes künstliches Gesicht zu generieren. Form, Aussehen und Ausdruck des Gesichts werden von den "Facial Description Parameters (FDP)" und "Facial Animation Parameters" (FAP) bestimmt. In MPEG-4 sind dann Methoden standardisiert wie FDP und FAP effizient kodiert und auch interpoliert (bei Datenverlust) werden können. In MPEG-4 Version 2 sind analoge Methoden für parametrische Körperbeschreibung und Animation enthalten.

Ein "2-D Mesh" ist eine Unterteilung einer 2-D planaren Region in polygonale Teile. Die Eckpunkte der Teile heissen "node points" des Meshs. MPEG-4 lässt nur Triangulierung zu. Ein 2-D dynamisches Mesh bezeichnet die geometrische Mesh Information und der Bewegung der node points in einer bestimmten Zeitspanne. Für das Mappen von Textur auf so ein Mesh werden die ursprünglich dreieckigen Teile des Mesh und der Textur durch die Bewegung verformt. Die Textur wird dann auf das entsprechende Teil des neuen Meshes durch "warping" angepasst. Wird die Möglichkeit der Abbildung auf eine affine beschränkt kann die Bewegung mit nur wenigen Parametern shr effizient gespeichert werden. Diese Art der Darstellung bietet einige interessante Features:



- Augmented Reality: Vermischung von computergenerierten Daten mit wirklichen bewegten Objekten
- Ersetzen eines natürlichen Videoobjekt durch ein synthetisches
- Verbesserte Kompression@low bit rates: nur die Textur wird komprimiert (und einige "Animationsparameter")
- Verbesserte Darstellung der Umrisse eines Objektes (Polygone statt Bitmaps)

1.3.2 Kompression von natüerlichen Videoobjekten

Die beliebige Form der Videoobjekte legt die Frage nahe, wie man Datenmaterial generieren kann, dass eine objektorientierte Segmentierung aufweist. Der MPEG-4 Standard legt hier nichts fest, jedoch sind folgende Varianten denkbar:

- Vollautomatische Segmentierung von vorhandenem Videomaterial: klappt nur bei einfachen head-and-shoulder Sequenzen, ansonsten ist das Problem der semantisch korrekten Segmentierung ein ungelöstes !
- Semiautomatische Segmentierung von vorhandenem Videomaterial mit manueller Segmentierung von Keyframes und temporal Tracking der Objektgrenzen.
- Neugenerierung von MPEG-4 konformen Videomaterial mit "blue-screen" ähnlichen Verfahren.

Bitstreamhierarchie

Ein MPEG-4 Bitstream liefert eine hierarchische Szenenbeschreibung: die komplette Szene heisst Visual Object Scene (VS) und enthält 2-D oder 3-D Objekte und die ensprechenden Enhancement Layers. Ein Video Objekt (VO) entspricht einem konkreten Objekt in der Szene, im einfachsten Fall ein rechteckiger Frame, oder eben ein Objekt beliebiger Gestalt, das Vorder- oder Hintergrund sein kann. Die nächste Ebene ist die der Video Object Layer (VOL): je nachdem ob das Objekt in skalierbarer oder nicht-skalierbarer Form gespeichert ist. Jedes Zeit Sample eines Video Objekts ist eine Video Object Plane (VOP), mehrere können zu einer Group of Video Object Planes (GOV) zusammengefasst sein.

Eine VOP kann verschieden kodiert sein. Im "klassischen" Fall enthält sie Bewegungsparameter, Gestaltinformation und Textur. Ebenso kann ein "Sprite" (siehe später) als VOP gespeichert sein. In MPEG-4 wird das 4:2:0 Chromasampling und die gutbekannte Makroblockstruktur unterstützt. Die unterschiedlichen Videoobjekte werden unabhängig verarbeitet und gespeichert.

Kodierung von Gestaltinformation

Im Fall der binären Variante wird die Gestalt durch eine binäre Matrix definiert. Die Gestalt einer VOP wird von einer rechteckigen Maske umschrieben, die 16x16 Blöcke (BAB - binary alpha block) aufgeteilt wird. Die Werte ausserhalt der VOP werden auf 0 gesetzt, die anderen auf 1 (oder 255). Es gibt dann Blöcke mit lauter gleichen Werten (transparent - alle 0, opaque - alle 255) und natüerlich sloche die an der Grenze der VOP liegen. Die BABs werden gespeichert durch arithmetische Kodierung und block matching mation estimation und DPCM Kodierung der Motion Vektoren. Im Fall des gray scale coding wird zusätzliche Transparenzinformation gespeichert, hier werden 8bpp verwendet. In diesem Fall wird mit lossy DCT coding gearbeitet (anstelle von arithmetischer Kodierung).

Bewegungskompensation

Motion Compensation verwendet Grundideen von früheren Standards, adaptiert an das VOP Konzept. Es gibt I-VOPs, P-VOPs und B-VOPs. Motion estimation wird natürelich nur für Makroblocks durchgeführt, die im die VOP umgebenden Rechteck liegen. Im Fall von Makroblöcken die ganz in der VOP liegen, wird alles so gemacht wie in MPEG-2 (es kann auch für jeden 8x8 Block ein Displacementvektor berechnet werden). Für nur teilweise in der VOP gelegene Makroblöcke wird die "modified block (polygon) matching technique" verwendet. Nur Pixelwerte inerhalb der VOP werden für die Differenzberechnung verwendet; wenn der Referenzblock ausserhalb liegt, werden Pixelwerte nach vorgegebenen Schema eingefüllt. Überlappende Motion Compensation ist ebenfalls erlaubt.

Textur Kodierung

Für Blöcke in der VOP wird alles gemacht wie in MPEG-1 oder 2. Andere Behandlung erfordern natürlich wieder Blöcke die an der Grenze der VOP liegen. Solche Blöcke werden mit Werten befüllt, bis sie "normale" Blöcke sind und dann entsprechend verarbeitet. Im Falle von B oder P-VOPs werden 0 befüllt, bei einer I-VOP wird der Durchschnittswert der VOP mit einigen zusätzlichen Änderungen verwendet. Um zusätzlich die Kodierungseffizienz zu erhöhen, können die Koeffizienten von benachbarten Blöcken zur Prediktion verwendet werden, entweder nur die DC oder auch wichtige AC Koeffizienten. Es gibt drei scan-Varianten, neben dem zig-zag scan auch alternate-horizontal und alternate-vertical scan (muss mit der Prediction der Koeffizienten übereinstimmen.

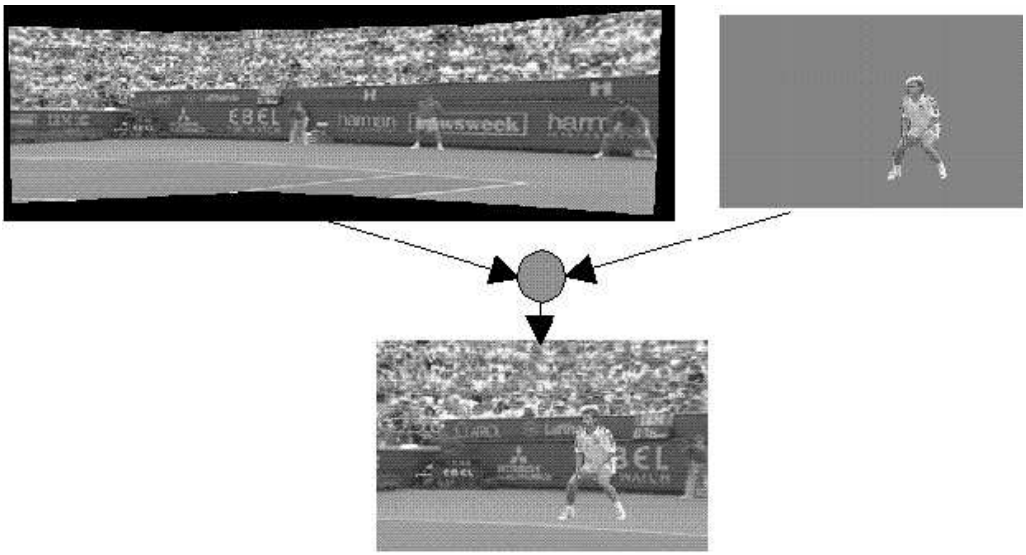
Eine Besonderheit in MPEG-4 ist, dass es einen eigenen Kompressionsmodus für sg. "statische Texturen" gibt, das sind solche, die auf 2-D (oder 3-D) Meshes gemapped werden. Diese Variante bietet *wesentlich* bessere Skalierbarkeit und ist mit einem Wavelet Verfahren realisiert. Das low-pass suband wird DPCM kodiert, die anderen subbands mit einem zerotree ähnlichem Verfahren. Da es in den früheren MPEG Standards keine Entsprechnung zu statischen Texturen gibt, konnte hier die Forderung nach Rückwärtskompatibilität unter den Tisch gefallen lassen werden.

Fehlerbehandlung, Sprites und Skalierbarkeit

Da auch an mobile Environments gedacht ist, ist die Möglichkeit der Fehlerbehandlung besonders wichtig.

- Resynchronisierung: eindeutige Marker werden in den Bitstream eingebettet, ab denen wieder (nach datenverlust) unabhängig dekodiert werden kann.
- Datenaufteilung: Daten für Bewegungsinformation und texturinformation werden getrennt, erlaubt effizienteren Einsatz von Error Concealment.
- Header extension Code: Zusätzlicher Einbau von redundanter Headerinformation (zum Dekodieren natürlich extrem wichtig)
- Reversible VLC: hier handelt es sich um Codewörter, die auch ein Rückwärtsdekodieren erlauben.

Ein "Sprite" ist ein besonders grosses Video Objekt, das während einer ganzen Szene vorhanden ist und meist grösser ist als der momentane Bildausschnitt. Klassisches Beispiel ist natüerlich ein Hintergrundsprite. Ein solches VO wird am Beginn einer Szene übertragen und während der Szene entsprechend modifiziert. Kodierung erfolgt wie bei I-VOPs.



Skalierbarkeit bezüglich der Auflösung im Bildbereich und Zeitbereich werden durch multiple VOL implementiert. Objektskalierbarkeit wird natürlicherweise durch das VO Konzept unterstützt. Im Bereich der temporalen Skalierbarkeit gibt es zwei Typen:

- Enhancement Layer verbessert nur die Auflösung eines Teils des Base-layers
- Enhancement Layer verbessert die Auflösung des gesamten Base-layers

1.4 Non-standard Methoden

Es gibt sehr viele nicht-standardisierte Lösungen im Bereich MJPEG, da häufig aus Kostengründen auf motion estimation/compensation verzichtet wird, z.B. das DVCR Industrial Consortium (Digital Video Camera Recorder): ist eine vollständige Spezifikation das das Band, Audio und Videokompression beinhaltet. Es gibt drei Modi, standard definition (SD), high definition (HD) und advanced TV (ATV), wobei SD und HD DCT basierte intraframe Kompression (mit VLC Kodierung) verwendet, wogegen ATV MPEG-2 ähnlich ist.

Im Bereich "echter" Videokompression kann man grundsätzlich drei Arten unterscheiden:

- 2 dimensionale Kompression mit Bewegungskompensation (wie MPEG-1-4, H.261/3)
- 3 dimensionale Kompression ohne Bewegungskompensation
- 3 dimensionale Kompression mit Bewegungskompensation

Verfahren die den letzten beiden Punkten entsprechen haben meist hohe Anforderungen an die Hardware u.a. bezüglich Speicherumfang und CPU-Leistung und sind daher für die Massenproduktion nicht geeignet obwohl sie oft bessere Kompressionsleistung aufweisen.

1.4.1 Fraktale Kompression

Ist nur sinnvoll, wenn nicht Fehlerbilder fraktal komprimiert werden (denn die dominierenden hochfrequenten Anteile sind extrem schlecht mit fraktaler Technologie exakt darstellbar), also z.B.

- Fraktale Kompression von 3 dimensionalen Datenblöcken; hier gibt es dann eine Vielzahl von Möglichkeiten zur adaptiven Unterteilung.
- "Fraktales" Mapping von Blöcken zwischen Frames (das ist Blockmatching mit Grauwertkorrektur) und eventuellem Kodieren der Fehlerbilder.

1.4.2 Wavelet-basierte Kompression

Ist grundsätzlich in einem analogen Setting wie MPEG verwendbar, da aber hier keien 8×8 Pixelblöcke verwendet werden ist beim Blockmatching ein Überlappen der Blöcke sehr empfehlenswert. Die Vorteile im Hinblick auf reine Kompressionsleistung sind geringer im Vergleich zu Stillbildkompression. Verwendet man ein 3 dimensionales Modell, ist es z.B. sinnvoll kürzere Filter in die Zeitrichtung zu verwenden um Fehler nicht über mehrere Frames zu verteilen. Sehr effizient bezüglich Kompressionsleistung ist natüerlich die Verwendung von 3 dimensionalen Zerotrees, aber natürlich auch sehr zeitaufwendig.

1.5 MPEG-7

Im Moment sind effiziente Lösungen zur Suche nach Informationen beschränkt auf Text-basierte Daten (siehe Internet Suchmaschinen), in Multimedia Datenbanken (MMDB) kann man bezüglich Farben, Texturen, und Formen suchen, Ähnlich ist auch denkbar für Audio, Kontext und Semantik fehlt allerdings völlig. Es geht nicht nur um Suche in MMDBs sondern auch z.B. um Auswahl aus einer grossen Menge von TV Sendern.

MPEG-7 hat nichts mit Kompressionsalgorithmen zu tun sondern ist eine *Multimedia Content Description Language*: es geht um die Definition einer standardisierten Menge von Deskriptoren, um Methoden neue Deskriptoren zu definieren und um Strukturen und Relationen zwischen Deskriptoren. Für dieses Ziel wird eine eigene Sprache entwickelt: *Description Definition Language*. MPEG-7 ist nicht beschränkt auf die Beschreibung von MPEG-1-4 kodierter Daten sondern kann auch an einen analogen Film oder ein gedruckted Bild "angehängt" werden.

Es gibt verschiedene Beschreibungsebenen

- low: Form, Grösse, Farbe, Position
- high: semantische Ebene, Kontext

und damit verbunden die Frage wie das extrahiert werden kann (mit oder ohne menschliche Hilfe).

Zusätzlich zur rein inhaltlichen Ebene kann auch noch folgendes erfasst werden:

- Form der Darstellung, z.B. wie kodiert, Datenmenge

- Bedingungen für Zugriff (Copyright, Preis)
- Klassifizierung (Kategorisierung, parental guidance)
- Links zu anderem relevanten Material
- Kontext (wann, wo, unter welchen Bedingungen aufgenommen)

MPEG-7 Verarbeitungskette: Feature Extraction (nicht definiert) => Beschreibung (MPEG-7) => Suche (nicht definiert)

Anwendungsbeispiele:

- Musik: einige Noten am Keyboard spielen und Musik angeboten bekommen, die irgendwie ähnlich ist, z.B. bezüglich der hervorgerufenen Emotionen
- Grafik: einige Linien auf einem sensiblen Schirm zeichnen und Bilder mit ähnlichen Grafiken, Logos angeboten bekommen
- Bilder: textuelle Eingabe ("Bild aus Terminator II wo .....") oder Angabe von Strukturelementen (z.B. Textur)
- Bewegung: Definition von einer Anzahl und der Bewegung zwischen Objekten (z.B. viele kleine Objekte fahren im Kreis)
- Stimme: Stimmsample von Pavarotti => Musik, Videos von/über ihn werden angeboten

Timeline: Call for Proposals Oktober 1998, Working Draft Dezember 1999, International Standard September 2001

Kapitel 2

Audio

Interessanterweise wird im Audiobereich schon seit langer Zeit auf die Eigenschaften der menschlichen Wahrnehmung mehr Rücksicht genommen als dies im visuellen Bereich der Fall ist. Es werden praktisch keine numerischen Fehlermasse verwendet um Kompressionsverfahren zu vergleichen. Aus diesem Grund ist es hier wichtig, die menschliche Lautwahrnehmung zu studieren.

2.1 Eigenschaften der menschlichen Lautwahrnehmung

Im Audio Bereich gibt es eigene Masszahlen die der Wahrnehmung entsprechen und die sich von den analogen "objektiven" Massen unterscheiden (Frequenz vs. Pitch, Intensität vs. Loudness). Eine Möglichkeit einen "transparenten" Koder zu definieren, ist zu sagen, hier muss das codierte Signal vom Original nicht unterscheidbar sein. Wesentlich schwieriger ist es zu sagen, welches von zwei kodierten Samples zu bevorzugen ist ("nicht-transparente Koder"). Methoden zum Coder-design:

- Feedforward Paradigm: das zu kodierende Signal wird analysiert und das mögliche Ausmass von Coding Noise so bestimmt.
- Feedback Paradigm: iterativ wird immer wieder bestimmt, welches von zwei (oder mehreren) codierten Signalen das bessere ist.

Loudness: Wird die Intensität eines Signals erhöht, steigt auch die Loudness, trotzdem sind die Terme nicht gleichbedeutend. Während Intensität mit einem Messinstrument gemessen werden kann, ist Loudness eine Wahrnehmungseinheit und kann nicht direkt gemessen werden sondern nur durch rating von Testpersonen bestimmt werden (daher wegen Subjektivität problematisch). Es gibt zwei Einheiten: Loudness level (in phon) und Loudness (in sone). Phon ist die Loudness eines 1 kHz Tones mit entsprechender Intensität in dB SPL (sound pressure level). Ein sone ist definiert als die Loudness eines 1 kHz Tones mit 40 dB SPL, ein Signal das als halb so laut empfunden wird hat 0.5 sone, etc. Zu beachten ist, dass die Messung durch Matching passiert wobei bei unterschiedlichen Frequenzen oder komplexen Signalen die Ergebnisse uneinheitlich werden können. Sone kann ebenfalls durch Matching bestimmt werden, wenn die Loudness Additivität ausgenutzt wird.

Pitch: Pitch verhält sich zu Frequenz ähnlich wie Intensität zu Loudness. Unter gewissen Bedingungen nimmt Loudness ab mit abnehmender Frequenz (bei konstanter Intensität) und Pitch nimmt ab bei steigender Intensität (bei konstanter Frequenz).

Hörschwelle: Schwellen werden nicht nur bestimmt durch Signal und Beobachter sondern auch durch die Methode der Messung. Die einfachste Definition von Hörschwelle ist die geringste Intensität die ein Hörer wahrnehmen kann. Das ist jedoch nicht adaquat, weil man die Wahrnehmung nicht direkt messen kann. D.h., es geht um die geringste Intensität, die einen Hörer zu einer Reaktion veranlasst. Es gibt zwar auf identische Signale nicht immer identische Responses, aber die Wahrscheinlichkeit eines positiven Responses nimmt mit der Signalintnesität zu. Man könnte hier die Schranke bei 50% Wahrscheinlichkeit fixieren. problematisch ist dabei, dass man keinerlei Kontrolle über die Kriterien der Testperson hat (z.B. ja wenn Signal sicher da ist, oder wenn es da sein könnte u.s.w.). Einen Ausweg bietet die Methode zwei Beobachtungsintervalle anzubieten in einem zufällig das Signal dargeboten wird. Der Beobachter muss entscheiden, in welchem das Signal war. Hier ist eine vernünftige Wahl der Schwelle bei 75% Wahrscheinlichkeit. Die Anzahl der Intervalle kann auch noch erhöht werden.

Unterscheidungsschwelle: oft bezeichnet als JND (just noticable difference), ist das Aussmass um das ein Attribut eines Signals sich verändern muss sodass ein Hörer die Änderung bemerkt (kann bezogen sein auf Frequenz, Intensität, Dauer, ...). Wieder gibt es die einfache Möglichkeit nur die Unterschiedlichkeit festzustellen, dann die Ausprägung der Unterschiedlichkeit (welches Signal hat höhere Frequenz) oder die Einteilung in N Intervalle und nur eines enthält ein unterschiedliches Signal.

Masking: hier kann zwischen gleichzeitigem und temporalem Masking unterschieden werden. Gleichzeitiges Masking ist ein Phänomen im Freuqenzbereich wo ein schwaches Signal durch ein gleichzeitiges stärkeres Signal unhörbar gemacht wird, wenn die beiden Signale ähnliche Frequenz haben. Die Maskierungs Schwelle ist diejenige Schwelle, unter der das schwächere Signal nicht wahrgenommen wird und hängt ab von der Frequenz, dem SPL, u.s.w. Das Phänomen des Masking ist ein wesentlicher Faktor beim Verstecken von Kodierungs Artefakten. Höhere Frequenzen werden leichter maskiert als niedere. Kodierungsrauschen ist nicht hörbar, solange SNR (m) grösser ist als SMR. Temporales Masking tritt auf wenn zwei Signale zeitlich nahe sind, wobei das stärkere das schwächere maskieren kann sogar wenn das Schwächere vorher kommt (wichtig um plötzlich auftretende grosse Quantisierungsfehler maskieren zu können) ! Premasking ist wesentlich kürzer als Postmasking.

Critical Bands: Das Hörsystem arbeitet wie eine Bandpass Filterbank. Der Frequenzraum ist in 25 kritische Frequenzbänder eingeteilt, die getrennt analysiert werden (energy detection) - diese Bänder sind 100 Hz breit unterhalb von 500 Hz und bis zu 5000 Hz für hochfrequente Signalteile. Zwei Signale die innerhalb solche Bänder liegen werden anders beurteilt als zwei Signale in verschiedenen Bändern. Die Eigenschaften im Bezug auf critical bands sind auch sehr wichtig im Hinblick auf das Design von Kodern, z.B. wegen des Zusammenhangs zwischen Bandbreite und Loudness und zwischen Bandbreite und Masking.

Einige Zusammenhänge: Loudness hängt stark von der Frequenz ab - am empfindlichsten ist die Region 2-3 kHz, Loudness wächst schneller mit der Intensität bei niederer Frequenz als bei höherer. Ist die Bandbreite eines Klanges innerhalb eines kritischen Bandes, ist die Loudness nicht von der Bandbreite abhängig. Wird die Bandbreite grösser, nimmt die Loudness mit zunehmender Bandbreite zu. In der Nähe der Hörschwelle kehrt sich diese Eigenschaft bei grosser Bandbreite um. Critical Duration: ist die Zeitdauer eines akustischen Signals, unterhalb der Loudness mit zunehmender Dauer ansteigt. Masking ist am stärksten ausgeprägt im Bereich von 400 Hz, aber nicht zu nahe bei 400. Bei geringer Intensität ist das Masking frequenzsymmetrisch, bei hoher Intensität gibt es starke Asymmetrie zugunsten der hohen Frequenzen. Ein Signal mir Bandbreite

innerhalb eines critical Bands erhöht das Masking bei sich erhöhender Bandbreite, ausserhalb eines critical Band gilt das nicht mehr.

2.2 Prinzip der low-bitrate Audiokodierung

Die Abhängigkeit der menschlichen Hörwahrnehmung von diversen Parametern wird ausgenutzt. Insbesondere werden Artefakte in Frequenzbänder geshiftet, wo sie nicht oder kaum wahrgenommen werden können. Dieses Shifting wird den Ergebnissen bezüglich SMR von Kurzzeit Spektralanalyse angepasst. Bei diesem Verfahren sind natürlich grundsätzlich Methoden im Fourierbereich von Vorteil, die hier eventuell keine getrennte Analyse und Kodierung durchgeführt werden muss.

2.3 Sprachkodierung

Die meisten Verfahren in diesem Bereich basieren auf linear predictive coding.

- ADPCM: die Differenz zwischen der adaptiv generierten prediction und dem Signal wird gespeichert. Der Prediktor ist rückwärtsadaptiv, d.h. die Koeffizienten werden nachadjustiert unter den Ergebnissen der Analyse der bereits kodierten Teilsignale. Ebenso ist die Quantisierung rückwärtsadaptiv. Die Variante *embedded* ADPCM ist für paketorientierte Netzwerke optimiert: das Netzwerk kann least significant bits bei Überlastung skippen, um packetloss zu vermeiden. GSM beutzt einen sog. Regular-pulse-excited long-term prediction linear predictive coder (RPE-LTP-LP Coder) bei dem das subgesampled Fehlersignal mit Kandidaten aus einem Codebook verglichen werden und die Adresse des besten übertragen wird.
- Analysis-by-Synthesis Coding: In einem Codebook gibt es eine Menge von Referenzsignalen die von einem Filter zusammengesetzt und skaliert werden. In einer Feedbackschleife werden alle möglichen Kandidaten getestet, die Adresse des ähnlichsten Signals wird übertragen. Der Filter und der Skalierungsfaktor weden während der Kodierung in Abhängigkeit von einer Fehleranalyse nachadjustiert. Der sog. CELP Coder benutzt zwei Codebooks: ein adaptives Codebook dass die am meisten verwendeten Signale beinhaltet und ein stochastisches Fehlercodebook, das dem eher zufällig strukturierten Restsignal angepasst ist.
- Füer Wideband Sprachkodierung (16 kHz Sampling statt 8) basiert auf dem Aufsplitten des Signals in zwei kritisch gesampled Subbands wobei das low Subband weniger quantisiert wird (6-bit statt 2-bit) und einem Narrowband Signal ähnelt. Die beiden Subband Sequenzen werden mit einem ADPCM Coder kodiert, Adaptation basiert auf einer Anpassung der 4 most significant bits (embedded coding).

2.4 Wideband Audio Kodierung

Wideband Audio Kodierung beruht in allen Varianten auf einer Form von Frequenzzerlegung die anschliessend quantisiert wird. Zwei Grundtypen können unterschieden werden, nämlich *adaptive transform coding* (ATC) und *subband coding* (SBC). Beide Verfahren benutzen eine Transformation die das Signal in Spektalkomponenten zerlegt. Haben diese Spektalkomponenten eine geringe Frequenzauflösung, spricht man von

subbands, andernfalls von Tranformationskoeffizienten. Beide Verfahren ermöglichen es auf natürliche Art auf Eigenschaften des Hörens einzugehen, da die im Frequenzbereich operieren. Ein wichtiger Punkt bei diesen Verfahren ist das Auftreten von sog. Pre-echoes (ähnlich zu Kopiereffekten auf Analogbänder). Folgt auf eine ruhige Periode heftiger Klang im selben zu verarbeitenden Block, so kommt es zu grossen Quantisierungsfehlern in diesem Bereich. Bei beiden Algorithmenklassen werden diese Fehler über einen grösseren Bereich gestreut, was besonders im low-bitrate Bereich zu hörbaren Fehlern führt. Solche Artefakte können maskiert werden, wenn die Zeitspanne ihres Auftretens kurz ist. Daher können sie reduziert werden wenn man kleine Blöcke des Signals verarbeitet. Andererseits ist die Sideinformation bei kleinen Blöcken sehr gross, sodass man die Technik des Window-switching verwendet: kleine Blöcke werden verwendet um das Auftreten von pre-echos zu kontrollieren, ansonsten werden grössere verwendet.

- **Subband Coding**: das Signal wird durch eine Filterbank bestehend aus M Bandpassfiltern geschickt, der Output wird downgesampled. Die Zusammensetzung ergibt wieder das Originalsignal wenn keine Quantisierung durchgeführt wird. Der klassische Vertreter ist MPEG-1 Audio Layer I und II. Durch eine FFT werden die Signal-to-Mask ratios bestimmt die dann im Quantisierungsprozess der Subbands umgesetzt werden. Wie im visuellen Teil von MPEG ist auch hier nur der Decoder standardisiert, sodass immer wieder verbesserte psychoakustische Modelle und Bit-allozierungsmethoden in den Encodern eingesetzt werden können. MPEG-1 hat die Modi Mono, Stereo, Dual (mit zwei separaten Kanälen für bilinguale Programme), und joint Stereo. Im letzten Modus werden Abhängigkeiten zwischen den einzelnen Kanälen ausgenutzt und bei hohen Freuqenzen wird nur die Summe L+R übertragen/gespeichert. Layer I benutzt 32 Subbands und fixe Blocklängen (8ms bei 48 kHz Sampling), die SMR wird mit 512 Punkt FFT ermittelt. Wird z.B. für Philips DCC verwendet. Layer II ist sehr ähnlich, benutzt 1024 Punkt FFT, Redundanzen in der Sideinformation bei benachbarten Blöcken wird ausgenutzt und eine feinere Quantisierung wird durchgeführt. Wird z.B. für ADR (Astra Digital Radio) verwendet.

- **Transform Coding**: für die Transformation wird hier meist FFT oder DCT verwendet die auf einen Datenblock angewendet werden. Diese Koeffizienten werden dann (entsprechen einem psychoakustischen Modell) quantisiert. Durch die Blockbildung kommt es zu Artefakten, die durch ein Überlappen der Datenblöcke kontrolliert wird (*modified DCT - MDCT*, die Technik heisst *time domain aliasing cancellation*. AC-2 Coding (Dolby) verwendet eine 512 Punkt MDCT und MDST (abwechselnd), die alle 256 Punkte durchgeführt wird. Benachbarte Koeffizienten werden dann so gruppiert, dass sie den critical Bands entsprechen. Ausser dieser Gruppierung verwendet der Coder kein psychoakustisches Modell. Die Bit-allocation wird direkt aus den Koeffizeinten ermittelt. PAC Coding (AT& T) verwendet L/R und M/S (mean/sum) Kodierung in einer signalabhängigen Weise.

- **Hybrid Coding**: Diese Kombinationen bieten den Vorteil, dass bei verschiedenen Frequenzen verschiedene Frequenzauflösungen verwendet werden können. nach einer Subband Zerlegung werden die Subbands mit einer MDCT weiter aufgespalten um hohe Frequenzauflösung zu erreichen. Der ATRAC Coder (Sony) verwendet drei Subbands und eine anschliessende MDCT mit 50% Überlappung. Dynamisches Window switching zwischen 512 Samples und 128 Samples (64 bei hohen Frequenzen) wird durchgeführt. Dieser Koder mit 44.1 kHz Samplingrate und 256 kb/s stereo Bitrate wurde für die Minidisc entwickelt (die ca. 20% der Speicherkapazität einer CD hat). DER Algorithmus in dieser Klasse ist momentan MPEG-1 Layer III (auch bekannt als MP-3). 32 Subbands werden durch eine 6 bzw. 18 Punkt MDCT mit 50% Überlappung weiterverarbeitet. Die 6 Punkt MDCT wird durchgeführt, falls Pre-echoes zu erwarten sind. Als einziger MPEG-1 Layer wird hier variable bit-rate coding und

non-uniform Quantisierung eingesetzt. Ausserdem wird ein iteratives Verfahren zur Optimierung von Quantisierung verwendet (iterative analysis-by-synthesis). Eutelsat SaRa verwendet dieses System.

2.5 Multichannel Audio Coding

Ein logischer Schritt in der Weiterentwicklung von Audio ist Multichannel Audio. Solche Systeme können nicht nur die üblichen Lautsprechnerkombinationen unterstützen (wie z.B. das 5.1 System), sondern auch eigene Kanäle für Seh- und Hörbehinderte bereitstellen. Die verschiedenen Kanäle werden in den effizienteren Algorithmen nicht unabhängig kodiert, es werden sogar Stereo irrelevante Teile des Signals identifiziert und als Mono kodiert. Der wichtigste Algorithmus in dieser Klasse ist MPEG-2. Es gibt hier zwei Algorithmen: einer davon ist forwärts kopatibel (d.h. er kann MPEG-1 Mono und Stereosignale dekodieren) und rückwärtskompatibel (d.h. ein MPEG-1 stereo Decoder kann einen MPEG-2 bitstream korrekt interpretieren), der andere weder noch.

- **Rückwärtskompatibles MPEG-2 Coding**: die Kompatibilität wird durch Matrizen erreicht, die ein Mischen der 5 Kanäle in 2 definieren (z.B. L0 = L + 0.7 C + 0.7 LS). Zwei der Signale (L0 und R0) werden in MPEG-1 Format kodiert. Die restlichen Kanäle werden so belegt, dass ein vollständiges 5 Kanal Signal wiederhergestellt werden kann. Diese werden in einem speziellen Feld des MPEG-1 Stroms transportiert, der vom MPEG-1 Decoder ignoriert wird.
- **MPEG-2 Advanced Audio Coding (AAC)**: Ist ein fortgeschrittener ATC mit einer 1024 Punkt MDCT, M/S coding und einem rückwärts adaptiven Prediktor im Transformationsbereich. Iterative Methode wird für Quantisierung und Bit-Allocation verwendet. Es gibt 3 Profiles: Main (mit dynamischen, adaptiven window switching), Low-complexity (ohne Prediction) und Sampling-rate-scalable (mit hybrider filterbank). Bis zu 46 Kanäle werden unterstützt, Standard 1997 finalisiert.
- **Dolby AC-3 Coding**: ist ein multichannel Coder mit der AC-2 Filterbank. Maskierung wird eingesetzt, wenn der Bitbedarf zu hoch wird, hierbei werden Koeffizienten ähnlicher Frequenz von verschiedenen Kanälen zu einem kombiniert. Wird in US HDTV Audio und bei DVD NTSC verwendet.

2.6 MPEG-4 Audio

MPEG-4 stell Tools für die Kodierung von Natural Sound (Sprache und Musik) als auch für die Generierung von synthetischen Sound aus parametrischen Modellen zur Verfügung. Die Representierung von synthetischen Sound kann von Text oder Instrument Beschreibungen stammen.

In MPEG-4 sind bereits existierende Standards zur Audiokodierung inkludiert, in diesen Fällen stellt MPEG-4 nur die Bitstream syntax und Dekodierungsverfahren als Tools zur Verfügung. An die jeweiligen Anwendungen und Bitraten angepasst, wird so ein einheitliches Framework für verschiedene Koder geschaffen. Für Sprachkodierung gibt es einen CELP Koder, für low-bitrate Sprachkodierung den sog. Harmonic Vector eXitation Coder (HVXC - hier wird nur die Frequenzcharakteristik eines Predictionfehlers kodiert), für generelle Audiokodierung wird MPEG-2 AAC und TwinVQ (die Kodierung des Spektrums erfolgt durch einen VQ) verwendet. Es kann bespielsweise ein base layer durch CELP erzeugt werden, der enhancement layer ist AAC kodiert.

Besonders interessant (und gänzlich analog zum visuellen Teil) ist die Standardisierung im Bereich synthetisierter Sound.

- **Text to Speech (TTS)**: Text oder ein Text mit bestimmten zusätzlichen Parametern (z.B. Phonem Dauer, Pitch Gestalt) kann verwendet werden um synthetische Sprache zu generieren. Es können Parameter erzeugt werden, die eine Synchronisation mit künstlichen animierten Gesichtern erlaubt. MPEG-4 standardisiert das Interface für die Operation eines TTS Systems (das TTSI), nicht aber das TTS System selbst !

- **Structured Audio Tools**: diese Tools dekodieren Input daten und produzieren Klänge. Die Dekodierung wird gesteuert von einer eigenen Synthesesprache genannt SAOL (Structured Audio Orchestra Language) die in MPEG-4 standardisiert ist. Diese Sprache wird benutzt um ein *Orchester* aus *Instrumenten* zu definieren (die im Bitstream geladen werden und nicht am terminal vorhanden sind). Ein Instrument ist eine kleine Menge von Signalverarbeitungsroutinen, die spezielle Klänge (ev. von Instrumenten) emulieren. Kontrolle über die Klangsynthese wird erreicht durch laden von Scripts/Scores, die verschiedene Instrumente aufrufen und zeitlich miteinander verknüpfen. Diese Beschreibung wird in der Sprache SASL (Structured Audio Score Language) geladen und kann auch benutzt werden, neue Klänge zu kreieren und existierende Klänge zu verändern. Für Synthese Prozesse die eine weniger starke und feine Kontrolle benötigen kann auch das etablierte MIDI Protokoll verendet werden (wurde im Synthisizer Bereich entwickelt). Auf diese Art können simple Audio Effekte wie Schritte oder Türenschlagen, aber auch wesentlich komplexere Audioevents wie Musik auf konventionellen Instrumenten oder futuristische Musik erzeugt werden. Für Terminals mit weniger Funktionalität oder weniger anspruchsvollen Anwendungen können auch Soundtabellen (wavetable bank format) und einfache Verarbeitungsmethoden (Filterung etc.) verwendet werden.

Kapitel 3

Multimedia Security

3.1 Watermarking

Die digitalen Medien haben nicht nur Vorteile gebracht. Teilweise geben die Erzeuger und Distributoren von digital vorhandenem Material nur zögerlich Zugriff auf ihre Daten, da ihr Urheberrecht nicht sichergestellt werden kann. Bisherige Problemlösungen sind meist Hardware-basiert (zB besondere Disketten-Spuren). Durch die Welle des elektronischen Kommerzes wird aber nach Software-Lösungen gesucht. Eine Möglichkeit besteht hier im Einbetten von Copyright- oder auch anderen Informationen in die Daten. Dieses Vorgehen wird als ,,Digital Watermarking" bezeichnete und soll das Vorhandensein eines unsichtbaren digitalen ,,Wasserzeichens" beschreiben.

Ein Vorteil der digitalen Medien ist die Möglichkeit, andere Daten in einem Hauptdatenstrom einzubetten. Darunter versteht man die Möglichkeit zusätzliche Daten für den Menschen unsichtbar beziehungsweise unhörbar einzubauen. Teilweise wurde das auch schon bei analogen Medien gemacht (siehe Geschichtliche Entwicklung) aber erst durch die digitale Technik kann man grössere Datenmengen einbetten.

Es gibt verschiedene Anwendungsbereiche wie

- Urheberschutz,
- Steganographie,
- Manipulationserkennung

Wenn es um den Schutz von Copyright-Informationen geht, unterscheidet man zwischen Watermarking und Fingerprinting. Beim Watermarking werden alle Kopien der Daten gleich behandelt. Ziel ist es Copyright- oder Kopierschutz-Informationen in die Daten einzubetten.

Beim Fingerprinting werden jedoch verschiedene Daten in verschiedenen Kopien eingebettet. Das kann vielerlei Anwendungen haben, zB kann man damit verfolgen, welche Lizenznehmer sich nicht an die Lizenzbedingungen halten und die Daten verteilen.

Zwischen der klassischen Steganographie und derartigen Copyright-Markierungen gibt es einen wichtigen Unterschied. Bei der Steganographie möchte man geheimhalten, dass irgendwie kommuniziert wird. Ein er-

folgreicher Angriff besteht also in dem Nachweis das Daten eingebettet sind. Wenn Copyright-Markierungen verwendet werden, ist es den Teilnehmern meist bewusst, dass sie markierte Daten haben, manchmal sind die Markierungen sogar sichtbar. Das Ziel von Angriffen ist also nicht ein Wasserzeichen zu erkennen, sondern es nutzlos zu machen oder ganz zu entfernen.

Für die Einbettung von Daten in einen Hauptdatenstrom gibt es noch weitere Anwendungsmöglichkeiten. Man kann zB zusätzliche Steuerinformationen (Schüttelkino, Lichteffekte, ...) einbauen, mehrere Sprachen unterstützen (mehrere Audio-Daten oder Untertitel) oder Referenzen einbauen. Aber auch für ,,pay-per-view"-Anwendungen sind derartige Techniken interessant.

Man kann aber zB auch mit einem beliebigen Algorithmus verschlüsselte Daten in einen anderen Datenstrom einbinden und damit unbemerkt kommunizieren. Die Daten können über einen geheimen Schlüssel so in die Hauptdaten eingebunden werden, dass sie weder statistisch erfasst noch vom Menschen erkannt werden können.

Einer der Gründe für die steigende Popularität von steganographischen Methoden liegt in den Verboten für starke Kryptographie in vielen Staaten. Durch das Verstecken der verschlüsselten Kommunikation in unschuldig wirkenden Daten, soll den Regierungen klar werden, dass sie ihre Verbote nicht durchsetzen können.

3.1.1 Geschichtliche Entwicklung

Der Oberbegriff für die Dateneinbettungs- und Watermarking-Technologien ist Steganographie. Das Wort kommt von den griechischen Wörtern stegano (versteckt) und graphos (schreiben) und heisst also so ca. ,,versteckt geschriebenes". Im Gegensatz zur Kryptographie, wo man den Inhalt von Nachrichten geheim halten möchte, will man bei der Steganographie Methoden finden um die Existenz einer Nachricht vor Beobachtern zu verstecken.

Die ersten Spuren von versteckter Schrift wurden in der 4000 Jahre alten Grabstätte eines ägyptischen Reichen am Rande des Nils gefunden. Die genaue Bedeutung der Symbol-Substitution wurde jedoch noch nicht gefunden. In Homer's ,,Iliad" wurde auch eine Anspielung auf versteckte Schrift gefunden. Steganographische Methoden als solche wurden zum ersten Mal in Geschichten von Herodotus einige Jahrhunderte später festgehalten. Er beschreibt, wie Griechen Nachrichten über die bösen Absichten von Xerxes versteckt unter der Wachsschicht der damaligen Schreibtafeln übermittelt haben. Der Begriff Steganographie wird erst um 1500 durch das Buch ,,Steganographia" von Trithemius geprägt.

Es wurden auch Hinweise auf steganographische und auch kryptographische Methoden in Chinesischer und Indischer Literatur gefunden.

Einige alte Techniken werden in der neueren Literatur wieder beschrieben. Ein Beispiel dafür sind sogenannte Semagramme, welche sehr geringe physikalische Unterschiede (zB unterschiedlich platzierte Buchstaben, feine Punkte oder Löcher) ausnutzen um Daten zu verstecken. Eine Methode geht auf Aenas den Taktiker zurück. Durch kleine Nadellöcher und Punkte werden Buchstaben markiert. Als Programm WitnesSoft gibt es diese Methode von Aliroo. Beim Ausdrucken werden feine Punkte ausgedruckt, welche nur durch einen hochauflösenden Scanner wieder erfasst werden können.

Ein frühes Beispiel für versteckte Copyright-Informationen ist von Bach bekannt. Er hat in vielen seiner Stücke seinen Namen B-A-C-H durch ein entsprechend häufiges Auftreten von Noten versteckt.

Von Spionen wurden geheime Nachrichten im Morse-Code in Bildern versteckt. Zum Beispiel konnte in einem Landschafts-Bild ein kurzes und ein langes Blatt für das entsprechende Zeichen stehen.

Die Politik ist natürlich eines der Anwendungsgebiete für Steganographie. Simmons Arbeiten zu dem Thema wurden zB aus folgendem Grund verfasst: Die USA und die USSR wollten die gegenseitige Ausrüstung mit Atomwaffen bei den Abrüstungsverträgen überwachen. Dazu sollten Sensoren in den Atomlagern untergebracht werden. Natürlich wollten sich beide Seiten sicher sein, dass die ,,gegnerischen" Sensoren nur Informationen über die Anzahl an Atomwaffen übermitteln, aber nichts über die Lage der Stützpunkte.

3.1.2 Anforderungen

Es ist klar, dass es für die verschiedenen Anwendungsmöglichkeiten auch verschiedene Anforderungen an die Dateneinbettung oder das Watermarking gibt.

- Robustheit: Manche Anwendungen laufen in einer fehler- und verlustfreien Umgebung ab, zB Daten die unverändert in einer Datenbank gespeichert werden.

  Es gibt jedoch auch Anwendungen die besondere Anforderungen an die Robustheit der Datenextraktion stellen. Zum Beispiel werden oft verlustbehaftete Datenkompressionen wie JPEG für Bilder, MPEG für Videos und MPEG Layer 3 für Audiodaten verwendet, um die benötigte Bandbreite zu verringern und damit die Effizienz zu steigern. Auch Veränderungen durch die Übertragung sollen sich nicht negativ auf die eingebetteten Daten auswirken.

  Aber auch bewusste Manipulationen von Benutzern müssen berücksichtigt werden. Es ist kein Problem für einen Anwender, mit einem Bildbearbeitungsprogramm zB die Schärfe zu verringern. Solche Operationen auf dem Hauptsignal haben natürlich auch Auswirkungen auf die eingebetteten Daten.

  Bei der Entwicklung von robusten Dateneinbettungs- und Watermarking-Verfahren wurden auch folgende Modifikationen berücksichtigt:

  - Hinzufügen von Rauschen
  - Filter (Hoch-, Tief- und Bandpass)
  - Komprimierung (JPEG, MPEG, wavelet)
  - Permutationen
  - Rotationen
  - Skalierung in Raum und/oder Zeit
  - Hinzufügen oder Entfernen von zusätzlichen Bildpunkten, Bildern eines Videos
  - Digital/Analog und Analog/Digital Wandlungen, zB auf Kassette aufnehmen und wieder digitalisieren oder ausdrucken und scannen.

  Es gibt bereits eigene Programme, die sich auf das Entfernen von Wasserzeichen oder eingebetteten Informationen spezialisiert haben. Diese Programme wenden geringe geometrische Transformationen an, wodurch zwar Wasserzeichen entfernt werden, der Benutzer jedoch keinen Unterschied merkt

  Für Image Verification (d.h. Überprüfung, ob an einem Bild etwas geändert wurde) darf ein watermark NICHT robust sein !

- Wahrnehmbarkeit: In den meisten Fällen soll es einem menschlichen Betrachter nicht auffallen, ob er ein Original-Bild oder ein Bild mit eingebetteten Daten betrachtet. Bei manchen Anwendungen ist ein Unterschied aber auch durchaus erwünscht, zB bei Vorschaubildern in niedriger Qualität.

  Um zu bestimmen, ob die Dateneinbettung oder das Wasserzeichen als unbemerkbar bezeichnet werden kann, wird folgendes Vorgehen verwendet: Bei einer zufälligen Serie von veränderten und unveränderten Signalen soll ein menschlicher Betrachter sagen, welches Signal die bessere Qualität hat. Wenn er dabei in ca. 50% der Fälle das veränderte Signal angibt, gilt die Dateneinbettung als unbemerkbar.

  Bei einer solchen Untersuchung müssen auch Transformationen berücksichtigt werden, die typischerweise auf das Signal wirken, bis es zum Benutzer kommt, zB Veränderungen durch Hochpässe oder eine Erhöhung der Bildschärfe. Das Signal mit den eingebetteten Daten soll nach den Transformationen die selben Artifakte hinterlassen wie das Originalsignal.

- Kapazität: Bei vielen Anwendungen, zB beim Einbinden einer Seriennummer oder einer Authoren-Identifikation, werden sehr geringe Bandbreiten benötigt. Die Daten können wiederholt eingebettet werden und sind dadurch robuster gegen Transformationen. Es ist klar, dass eine möglichst hohe Einbettungskapazität bei gleichzeitig geringer Störung des Signals eine wünschenswerte Eigenschaft ist.

- Sicherheit: Bei vielen Anwendungen muss es für einen unbeteiligten Beobachter unmöglich sein das Vorhandensein von versteckten Daten zu erkennen. Die Sicherheit wird bei der Dateneinbettung genauso aufgefasst wie in der Kryptographie. Auch wenn der verwendete Algorithmus bekannt ist, soll es unmöglich sein zu erkennen ob Daten eingebunden wurden, wenn ein geheimer Schlüssel nicht bekannt ist. Natürlich soll es auch nicht möglich sein die Daten zu extrahieren oder zu verändern.

  Bei der versteckten Kommunikation werden die Daten meist auch noch verschlüsselt, bevor sie in ein anderes Signal eingebettet werden.

- Vorhandensein des Originalsignals beim Detekten: Bei manchen Anwendungen kann das Originalsignal verwendet werden, um die Daten aus dem empfangenen Datenstrom zu extrahieren, zum Beispiel bei der Verfolgung von Kopien oder dem Copyright-Schutz. In diesem Fall ist die Datenextraktion leicht, da man zuerst die Transformationen des empfangenen Signals rückgängig machen kann und erst dann die Daten extrahiert.

  Wenn das Originalsignal nicht vorliegt, ist die Extraktion schwieriger. Durch die verschiedensten Transformationen auf der Übertragungsstrecke kann das Auffinden der eingebetteten Daten sehr schwierig werden, weshalb meist weniger Daten eingebettet werden können.

3.1.3 Methoden

Wenn man nachdenkt, welcher Teil von zB einem Bild für den Benutzer am unauffälligsten zu verändern ist, kommt man sehr schnell auf die Idee, die Daten in den unbedeutendsten Bits der Farbinformation zu verstecken. Bei einem 24-bit Truecolor-Bild unterscheidet sich der Farbwert nicht besonders, wenn zB in jeder der drei Farbkomponenten ein Bit umgeändert wird. Bei einem Index-Format wie zB GIF, muss man vielleicht die Palette umordnen um in den LSBs Daten zu verstecken. Diese Methode ist aber auch am leichtesten zu entdecken und zu entfernen. So gut wie jeder Filter wird die meisten LSBs eines Bildes ändern und verlustbehaftete Kompression überstehen sie auch nicht unbeschadet.

Ausgefeiltere Methoden verwenden einen durch einen PRNG bestimmten Weg durch das Bild und bestimmen zusätzlich, ob es stören würde, wenn an dieser Stelle Pixel verändert werden. Dazu wird bestimmt, ob der Punkt in einer grossen monochromen Fläche oder nahe einer starken Veränderung ist, also ob die Varianz sehr klein oder sehr gross ist.

Spatial Domain Methoden

Verfahren zum Einbetten von Copyrightinformation

- Algorithmus von Kutter: Ein Watermark Bitstring wird im Blauen Kanal eingebettet durch Addition eines Bruchteiles des Luminance Channels. Die Einbettung geschieht redundant, d.h. der String wird mehrmals eingebettet.

$$b'(x,y) = b(x,y) + \alpha l(x,y) \text{ if } s_i = 0$$
$$b'(x,y) = b(x,y) - \alpha l(x,y) \text{ if } s_i = 1$$

  α steuert die Einbettungsstärke. Zum Auslesen der Watermarkbits wird eine sternförmige Umgebung eines Pixel betrachtet und daraus eine Prediction für das zentrale Pixel berechnet. Für jedes Watermarkbit sind nun durch die redundante Einbettung mehrere Predictions und veränderte Werte vorhanden was für die Rekonstruktion verwendet wird.

- Algorithmus von Bruyndoncks: Ein Watermark Bitstring wird Bildblöcken eingebettet. Dazu sind folgende Schritte nötig:
  - Klassifikation der Pixel eines Blocks in zwei Zonen homogener Helligkeit.
  - Unterteilung jeder Zone in Kategorien die durch ein vorgegebenes Muster definiert werden.
  - Manipulation der Helligkeitsmittelwerte für jede Kategorie in jeder Zone.

  Im Detail geht das wie folgt: Die Pixel in den Blöcken müssen in zwei Zonen homogener Helligkeit aufgeteilt werden. ES werden zwei Typen von Blöcken unterschieden: a) Blöcke mit hartem oder progressivem Kontrast und b) Blöcke mit Noise Kontrast.

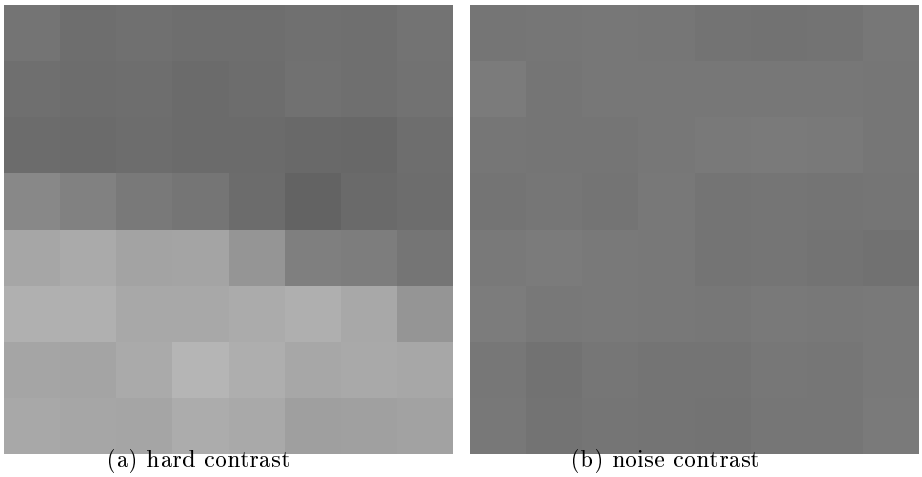
  Der Block mit Noise Kontrast kann nicht sofort in zwei Zonen eingeteilt werden - der Block mit hartem Kontrast kann sofort behandelt werden. Die Zonen müssen nicht zusammenhängend sein und müssen auch nicht gleich viele Pixel enthalten. Um die Klassifikation durchzuführen werden die helligkeitswerte im Block sortiert, im entsprechenden Graphen ist der Punkt mit maximaler Steigung ein Kandidat um die Pixel in zwei Zonen zu splitten.

  Ist diese Steigung größer als eine Schranke, wird hier geteilt. Andernfalls werden die Zonen willkürlich definiert und werden so gewählt dass sie gleich viele Pixel haben.

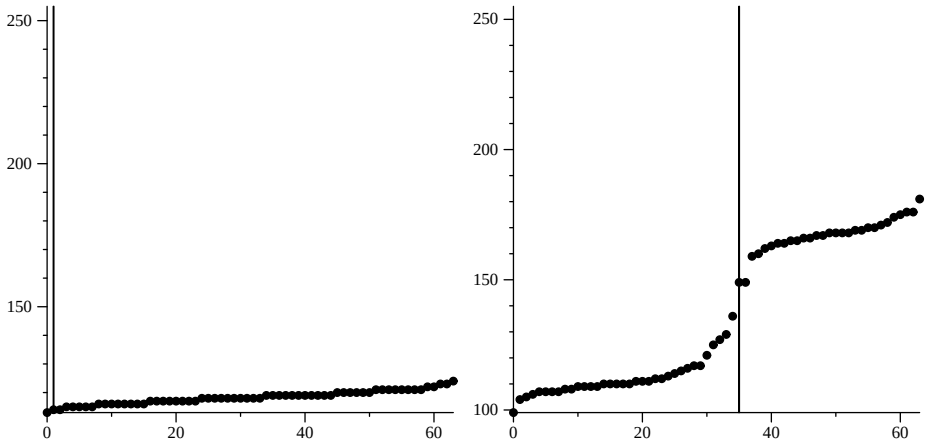
  Weiters werden nun muster verwendet um die beiden Zonen in unterschiedliche Kategorien einzuteilen. Ein unterschiedliches Muster wird für die beiden Zonen verwendet. Die Muster sind geheim und wechseln von Block zu Block (hier kommt offensichtlich ein zusätzliches Sicherheitskonzept ins Spiel).

  Nun sind die Pixel in den Blöcken in 5 Mengen unterteilt: die Weggelassenen (bei stark progressivem Kontrast), die restlichen sind in Zone eins oder zwei in Kategorie A oder B. Die Pixel in jeder Zone werden gezählt ($n_{1A}, n_{1B}, n_{2A}, n_{2B}$) und die Mittelwerte der Helligkeiten werden berechnet: $l_{1A}, l_{1B}, l_{2A}, l_{2B}$. Durch die Art der Konstruktion ist sichergestellt dass:

$$l_{1A} < l_{2A}$$



(a) hard contrast (b) noise contrast



Block no 70 is noisy Block no 17 is hard or progressive

$$l_{1B} < l_{2B}$$

Nun werden die Watermarkbits wie folgt eingebettet:

$$\left.\begin{array}{l} l'_{1A} - l'_{1B} = +\alpha \\ l'_{2A} - l'_{2B} = +\alpha \end{array}\right\} \text{if} \quad s = 1$$

$$\left.\begin{array}{l} l'_{1A} - l'_{1B} = -\alpha \\ l'_{2A} - l'_{2B} = -\alpha \end{array}\right\} \text{if} \quad s = 0$$

Gleichzeitig soll aber der Helligkeitsmittelwert in jeder Zone konstant gehalten werden:

$$\frac{n_{1A} l'_{1A} + n_{1B} l'_{1B}}{n_{1A} + n_{1B}} = l_1$$

$$\frac{n_{2A} l'_{2A} + n_{2B} l'_{2B}}{n_{2A} + n_{2B}} = l_2$$

A	A	B	B	A	A	B	B
A	A	B	B	A	A	B	B
B	B	A	A	B	B	A	A
B	B	A	A	B	B	A	A
A	A	B	B	A	A	B	B
A	A	B	B	A	A	B	B
B	B	A	A	B	B	A	A
B	B	A	A	B	B	A	A

(e) Grid for zone 1

B	B	B	B	A	A	A	A
B	B	B	B	A	A	A	A
B	B	B	B	A	A	A	A
B	B	B	B	A	A	A	A
A	A	A	A	B	B	B	B
A	A	A	A	B	B	B	B
A	A	A	A	B	B	B	B
A	A	A	A	B	B	B	B

(f) Grid for zone 2

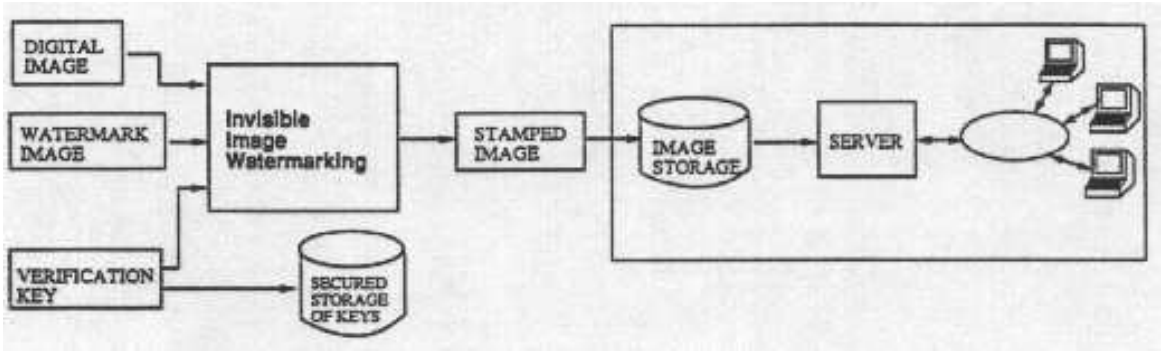
Um das zu erreichen, wird die Helligkeit jedes Pixel in einer Zone um den gleichen Wert verändert, z.B. werden die Pixel in Zone 1, Kategorie A um den Wert $|l'_{1A} - l_{1A}|$ verändert.

Um ein Watermarkbit auszulesen wird der analoge Algorithmus angewendet. Anstelle die Mittelwerte aber zu manipulieren werden sie berechnet und die Unterschiede bestimmt:

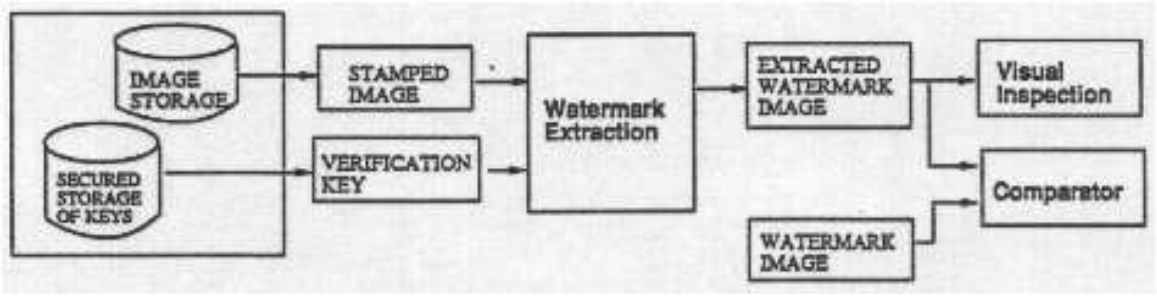
$$s''_k = \begin{cases} 0 & \text{if } l''_{1A} - l''_{1B} < 0 \text{ und } l''_{2A} - l''_{2B} < 0 \\ 1 & \text{if } l''_{1A} - l''_{1B} > 0 \text{ und } l''_{2A} - l''_{2B} > 0 \end{cases} \tag{3.1}$$

Nun könenn Standardmethoden verwendet werden um das rekosntruierte Watermark S'' mit dem Original S zu vergleichen.

IBM Verfahren zur Image Verification Ein Algorithmus ganz anderer Funktionsweise ist der im Folgenden beschriebene für Image Verification. Zum Einbetten ist ein zu markierendes Bild, das Wasserzeichen (in diesem Fall ein binäres Bild, ist es zu klein, kann es durch Kachelung auf die Grösse des Originalbildes gebracht werden) und ein Key (verification key) notwendig. Die Aufbewahrung der Keys erfolgt in einer geschützten Datenbank - nur mit diesem ist die Extrahierung des Wasserzeichen uns somit eine Verifizierung möglich. Das markierte Bild (stamped image) kann nun öffentlich zugänglich gemacht werden.

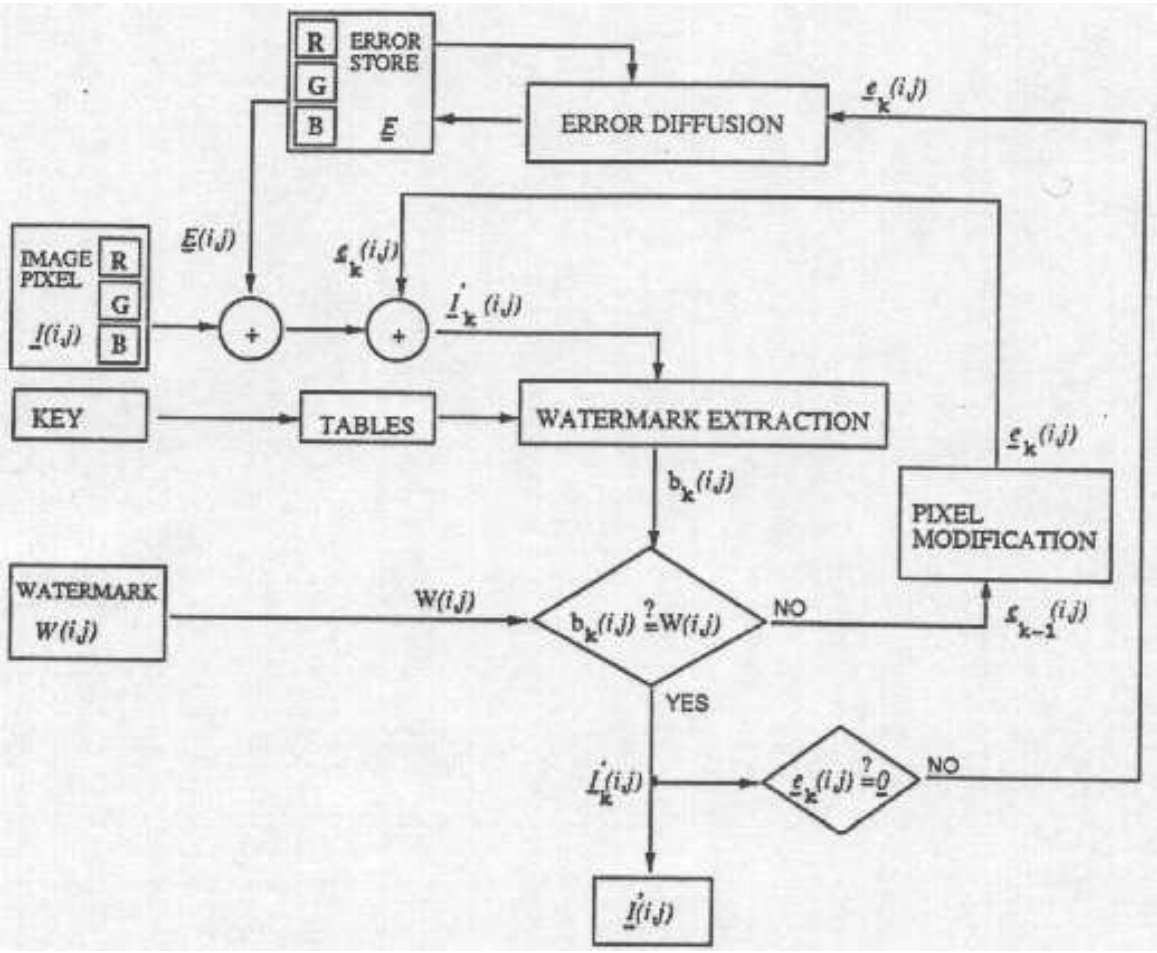


Zur Verifizierung des Bildinhaltes ist der verification key und das markierte Bild (stamped image) notwendig - das Originalbild wird also zur Kontrolle nicht mehr benötigt ! Der Extrahierungsprozess liefert ein Bild, das mit dem verwendeten watermark image übereinstimmen sollte. Ist dies nicht der Fall, wurde das Bild manipuliert. Die Kontrolle kann visuell oder rechnerunterstützt erfolgen.



Die folgende Graphik zeigt den Einbettungsvorgang für ein einzelnes Pixel, wobei W(i,j) das Watermark und I(i,j) das Originalbild, (i,j) die Pixellokation, $b_k(i,j)$ ein extrahiertes Watermark Pixel, $e_k(i,j)$ die Differenz zwischen W und b_k und E(i,j) der verteilte Fehler für eine best. Farbe sind.

Es wird iterativ Pixel nach Pixel (von links oben nach rechts unten) bearbeitet. Grundlegend ist, dass das watermark image nicht gezielt in das Originalbild eingebettet wird, sondern das Originalbild solange verändert wird (pixel modification) bis das extrahierte watermark image b_k mit dem gewollten W übereinstimmt.



Folgende Operationen werden dabei durchgeführt:

- Watermark Extraction: Die Aufgabe dieser Funktion ist es, des Farbwert des behandelten Bildpunktes

in einen schwarz/weiss-Wert abzubilden. Die Funktion basiert auf einem Key (Tabellen), der z.B. zufaellig erzeugt werden kann. Die Ergebnisse jedes Farbkanals werden mit XOR verknüpft.

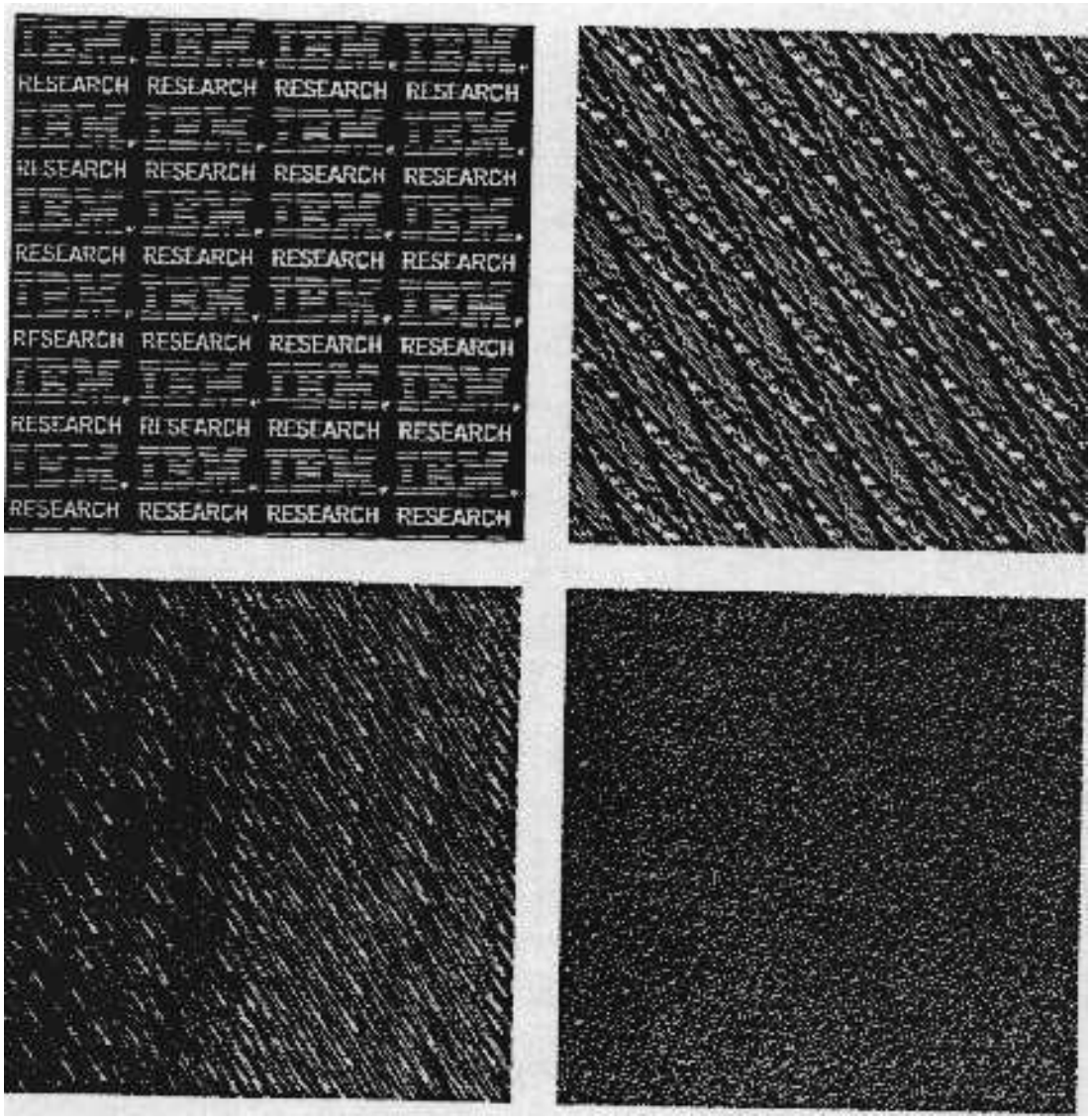
- Pixel Modification: Dies ist ein iterativer Prozess der so oft wiederholt wird, bis das extrahierte Wasserzeichen dem Gewünschten entspricht. In jeder Iteration wird einer der drei Farkanäle mit +1 oder -1 verändert. Dies soll möglichst zufällig erfolgen.

- Error Diffusion: Einbettung des Watermark Image verursacht Abweichungen vom Original. Das Ziel ist es, die durchschnittlichen Farbwerte der Pixel beibehalten um damit den Farbeindruck zu erhalten. Bei Initialisierung ist E(i,j) = 0 für alle (i,j). Der jeweils aktuelle Farbwert ist P(i,j) = I(i,j) + E(i,j) d.h. der Farbwert des originalpixels addiert mit mit der bereits für diesen Pixel durchgeführten Fehlerdiffusion. I'(i,j) wäre der gewünschte Output des Prozesses, damit W(i,j) = $b_k(i,j)$, das extrahierte Bild also mit dem Watermark übereinstimmt. Der Fehler der gemacht werden muss um das zu erreichen ist also $\delta(i,j)$ = P(i,j) - I'(i,j). Da dieser Fehler nicht erwünscht ist wird er lokal ausgeglichen um das Ziel der Erhaltung des durchschnittlich gleichbleibenden Farbwertes zu erreichen. Der Fehler wird bei dem Nachbarpunkten rechts und unten behoben (also jeweils die Hälfte von $\delta(i,j)$ wird dazuaddiert.

Das Verfahren hat sehr gute Eigenschaften:

- sehr schnell
- sehr sensibel auf Veränderungen, nicht nur LSBs verwendet !
- Herausfiltern der Watermark nur mit den drei Tabellen möglich (verification key).

- Eine mögliche Attacke ist, die Tabellen durch Probieren zu erstellen, wenn das Watermark bekannt oder visuell sinnvoll ist (ist v.a. bei GW Bildern heikel, da die Tabellen nur 256 Einträge haben). Daher sollte das Wasserzeichen möglichst nicht bekannt oder visuell erkennbar sein. Eine Möglichkeit das zu erreichen ist "chaotic mixing": Es werden nach einer ganz genau bestimmten Methode Pixel des Originalbildes an eine andere Stelle gesetzt. Das Firmenlogo ist nun nicht mehr klar erkennbar - für den Eigentümer ist es einfach nach der watermark extraction das mixing wieder rückgängig zu machen (die genaue Kenntnis der Transformation kann als Schlüssel gesehen werden !)



Transform Domain Methoden zur Copyrighteinbettung

Es gibt hier zwei grundsätzlich unterschiedliche Methoden, additives sowie quantize and replace Einbetten. Im zweiten Fall wird Watermark Information durch eine unterschiedliche Wahl von Quantizern eingebettet, die hier beschriebenen Algorithmen gehören der ersten Klasse an. Das Wasserzeichen besteht aus einer normalverteilten Folge von n Zahlen s_i mit Mittelwert 0 und Varianz 1. Um nun ein Element der Water-

markfolge s_i in einen Koeffizienten c_i einzubetten wird wie folgt vorgegangen (wobei c_i' der gewatermarkte Koeffizient ist und α die Einbettungsstärke ist):

$$c_i' = c_i(1 + \alpha s_i) \tag{3.2}$$

Die entsprechende Formel zur Extraktion der Watermarkinformation läßt sich leicht ableiten (wobei c_i^* den extrahierten koeffizienten bezeichnet und s_i^* das extrahierte Watermark Element):

$$s_i^* = \frac{c_i^* - c_i}{\alpha c_i} \tag{3.3}$$

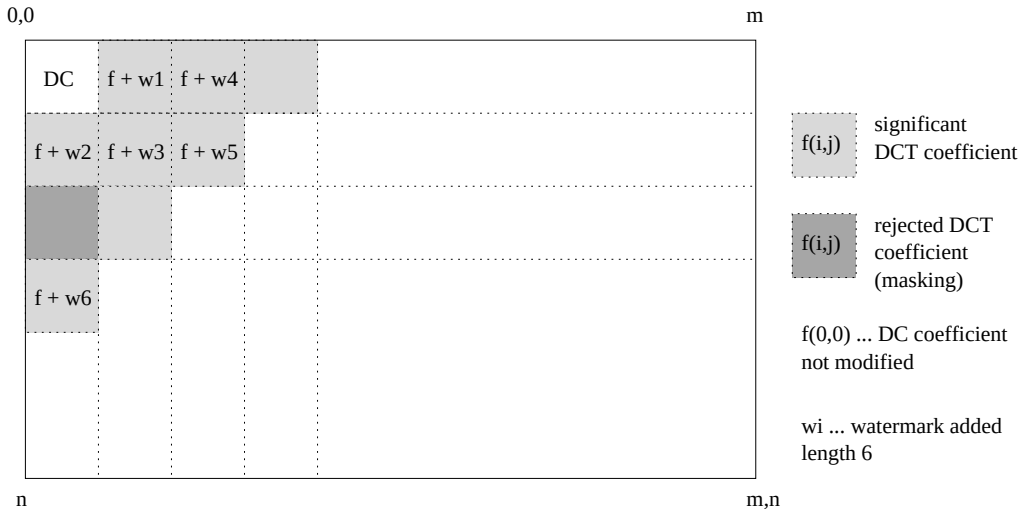
Im nächsten Schritt wird die extrahierte Wasserzeichenfolge s^* mit der Originalversion s verglichen (normierter Korrelationskoeffizient):

$$C(s, s^*) = \frac{\sum(s_i - \overline{s_i})(s_i^* - \overline{s_i^*})}{\sqrt{\sum(s_i - \overline{s_i})^2}\sqrt{\sum(s_i^* - \overline{s_i^*})^2}}$$

wobei $\overline{s_i}$ den Mittelwert der Folge s_i bezeichnet.

DFT and DCT Domain DER klassische Algorithmus zur Einbettung von copyright-schützenden Watermarks ist der von Cox (NEC Research, MIT): zu den 1000 grössten DCT-Koeffizienten $f(m, n)$ wird eine eben so lange um 0 normalverteilte Folge von Zufallszahlen w_i addiert:

$$f'(m, n) = f(m, n)(1 + \alpha w_i)$$



Für die Wiedergewinnung braucht man hier allerdings das Originalbild (und dieses DCT transformiert, Koeffizientenweise wird die Watermark dann wieder extrahiert). Die Idee ist hier, die Watermarking-bits in Bildregionen einzubetten, die wichtig sind und bei Kompression nicht berührt werden.

Weitere Algorithmen wären die von Koch und Benham: in 8x8 Block DCT Koeffizienten wird ein Watermark eingebettet, indem eine Größenreihenfolge auf ein zufällig gewähltes Tupel/Tripel von Koeffizienten verlangt wird (d.h. die Größe wird modifiziert). Bei der Benhamvariante werden diejenigen Blöcke nicht verwendet die ganz glatt sind oder die nur wenige dominierende Kanten aufweisen (das wird durch Untersuchung der DCT Koeffizienten festgestellt).

Wavelet Domain Verfahren Der Algorithmus von Corvi ahmt den Algorithmus von Cox im Waveletbereich nach. Die DWT wird bis zu einer bestimmten Zerlegungsstufe iteriert, dann werden alle approximation-subband Koeffizienten manipuliert. Zuerst wird der Durchschnitt der Approximationskoeffizienten $c_{l,i}$ berechnet: $\overline{c_l}$. Um nun ein Watermarkbit s_i einzubetten, wird der Koeffizient $c_{l,i}$ ersetzt durch:

$$c'_{l,i} = \overline{c_l} + (c_{l,i} - \overline{c_l})(1 + \alpha s_i)$$

Um zu verifizieren ob das Bild die Watermark s enthält, wird es entsprechend weit zerlegt, Durchschnitt $\overline{c}^*_l$ und Varianz $\sigma(c^*_l)$ der Koeffizienten berechnet und das rekonstruierte Watermarkelement s^*_i wird bestimmt wie folgt:

$$s^*_i = \frac{(c^*_{l,i} - \overline{c^*_l})\frac{\sigma(c_l)}{\sigma(c^*_l)} - (c_{l,i} - \overline{c_l})}{c_{l,i} - \overline{c_l}}$$

Berücksichtigung von Mittelwert und Varianz machen den Algorithmus robust gegen Kontrast und Helligkeitsveränderungen.

Der Algorithmus von Wang basiert auf der Einbettung der Watermarkinformation in die n most "significant" Detailkoeffizienten. Die Selektion dieser Koeffizienten ist iterativ. Für jedes Subband k auf Zerlegungsstufe l wird der größte Koeffizientenwert $\hat{c}_{k,l}$ bestimmt. Für jedes Subband wird ein Threshold $T_{k,l}$ bestimmt:

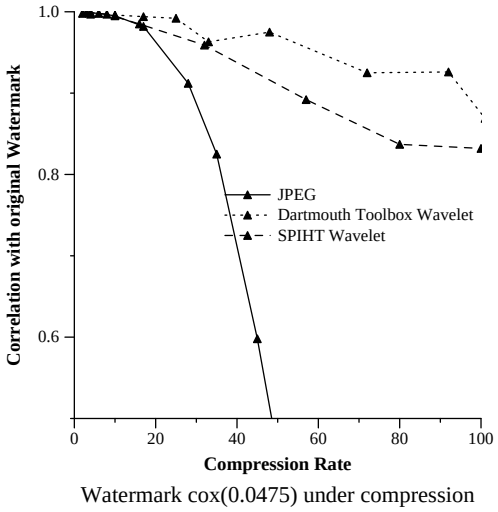
$$T_{k,l} = \frac{1}{2}\beta_{k,l}\hat{c}_{k,l}$$

wobei $\beta_{k,l} \in (0, 1]$ subbandabhängige Parameter sind. Eine Menge nicht-markierter Koeffizienten U wird initialisiert die alle Koeffizienten außer den Approximation-Koeffizienten enthält. Dann wird die folgende iteration ausgeführt, bis alle Koeffizienten selektiert sind:

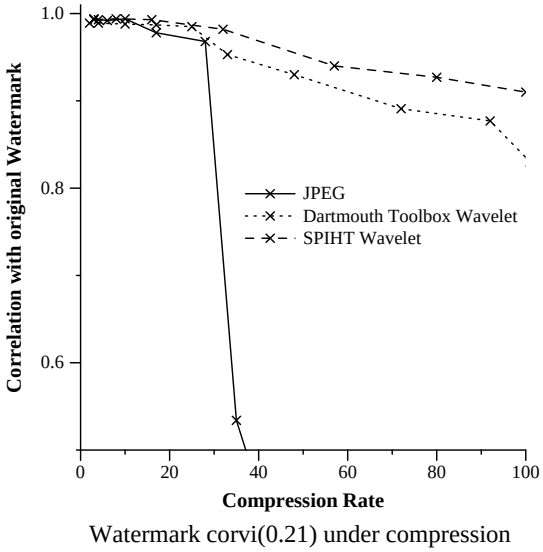
1. Wähle das Subband mit dem maximalen Wert für $T_{k,l}$.
2. Die Koeffizienten des ausgewählten Subbands die noch in U sind und größer als $T_{k,l}$ sind werden aus U entfernt und gewatermarkt.
3. $T_{k,l}$ wird halbiert.
4. Sind noch Watermarkelemente nicht eingebettet, nochmals iterieren.

Für das tatsächliche Einbetten wird die folgende Formel verwendet:

$$c'_{k,l,i} = c_{k,l,i} + \alpha\beta_{k,l}T_{k,l}s_i$$



(g) Cox



(h) Corvi

3.2 Videoverschlüsselung, DVD

Im Bereich Videoverschlüsselung (hier liegt der Schwerpunkt auf MPEG Videos) gibt es einige verschiedene Ansätze, die je nach Anwendung verschidenen Sicherheitsstufen und Komplexitätsgrade aufweisen. Durch die bekannte Struktur des MPEG Bitstreams (z.B. der Anordnung der DCT Koeffizienten und des Scanmusters) gibt es durchaus Möglichkeiten für Attacken wenn mit zu einfachen Methoden gearbeitet wird.

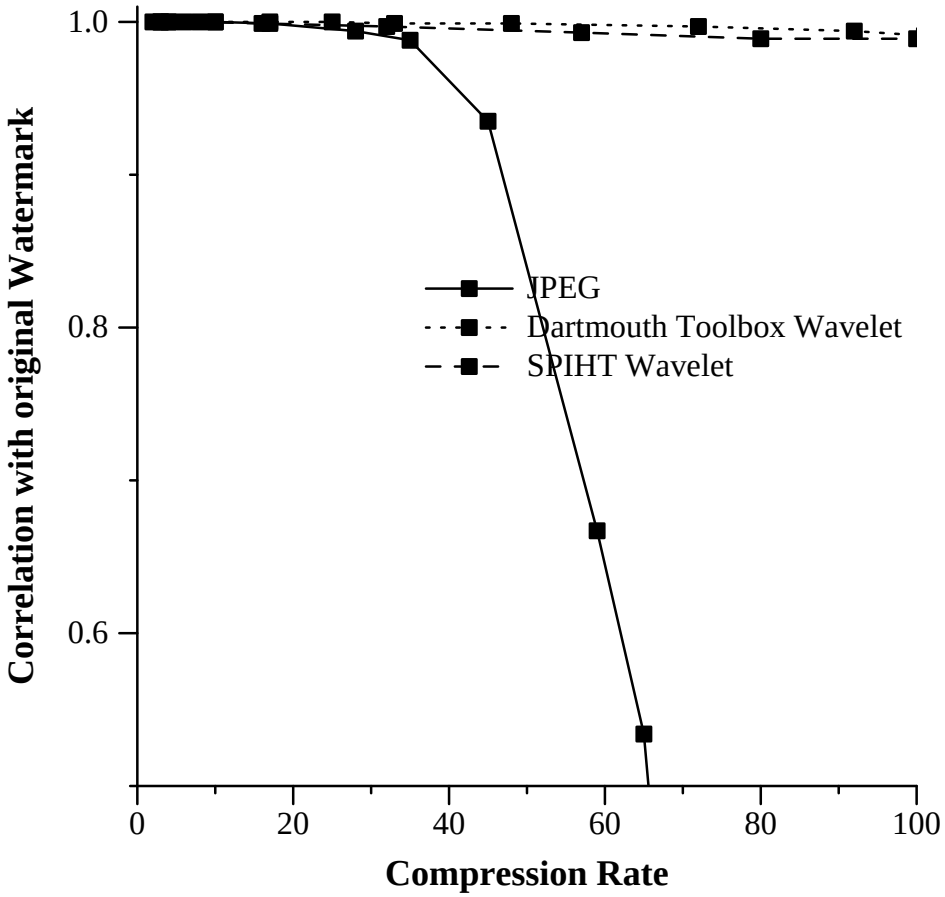
3.2.1 Naive Algorithm

Dabei kommen bekannte Verschlüsselungsalgorithmen zu Einsatz. Auf die Besonderheit des MPEG Datenstromes wird keine Rücksicht genommen.

- Sicherheit: So sicher wie das verwendete Verfahren
- Geschwindigkeit: Langsam, da alles verschlüsselt wird
- Datenstromgrösse: Keine Änderung

3.2.2 Pure Permutation Algorithm

Dabei wird der Datenstrom aus der MPEG Codierung mittels eines Permutationsschlüssels permutiert. Diese Methode ist jedoch anfällig für die known-plaintext Attacke, da man nur den Plaintext mit dem



Watermark wang(0.275) under compression

Ciphertext vergleichen muss. Aufgrund der gleichmässigen Verteilung der Daten im Strom sollte 1 I-frame zum Brechen genügen; dieser kann z.B. aus einem bekannten Firmenlogo stammen.

- Sicherheit: gering
- Geschwindigkeit: schnell
- Datenstromgrösse: Keine Änderung

3.2.3 Zig-Zag Permutation Algorithm

Das bekannte Zig-zag Muster wird hierbei durch eine Permutation verändert. Diese Methode ist anfällig für die known-plaintext Attacke und die ciphertext-only Attacke. Bei der known-plaintext Attacke kann durch einfachen Vergleich die Permutation gefunden werden. Die ciphertext-only Attacke ist auch erfolgreich, da sich die grossen Werte (DC und unteren ACs) in der linken oberen Ecke sammeln. Ein richtiges Raten der ersten fünf Werte liefert schon ein brauchbares Bild.

- Sicherheit: gering

- Geschwindigkeit: schnell
- Datenstromgrösse: grosser Zuwachs, da durch anderes scan-Muster die RLE behindert wird.

3.2.4 Selective Algorithm

Dies sind alle Algorithmen, die sich der MPEG Struktur bedienen um nur gewisse, ausgewählte Teile zu verschlüsseln. Basis hierfür ist, dass ein Datenstrom nutzlos ist wenn die I-frames fehlen; daher werden nur diese verschlüsselt, der Rest (P, B-frames) ist dann unbrauchbar. Dieser an sich brauchbare Ansatz bietet einige Probleme: Zum einen können I-frames Teile in P- und B-frames eingebettet sein und so zu sichtbaren Teilen führen. Abgesehen davon kann man bei Videos, wo sich der Hintergrund relativ wenig ändert auch ohne I-frames gewisse Bewegungen erkennen. Die Sicherheit dieses Verfahrens kann durch vermehrte Verwendung von I-frames gesteigert werden, da so mehr Bildteile verschlüsselt übertragen werden. Jedoch steigt so auch die Grösse des Datenstroms unnötig an.

SECMPEG nutzt die oben beschriebene Basis. Für die Verschlüsselungsteile wird DES oder RSA verwendet. Es erfolgt eine Einteilung in 4 Sicherheitsebenen:

1. Header verschlüsseln
2. Header, DC und die ersten ACs von I-frames verschlüsseln
3. Alle I-frame Blöcke verschlüsseln
4. Allses verschlüsseln (naiver Alg.)

Nachteil von SECMPEG ist dass es nicht kompatibel ist zu MPEG-2, d.h. man braucht eigene encoder/decoder Einheiten.

- Sicherheit: je nach Sicherheitsstufe
- Geschwindigkeit: schnell
- Datebstromgrösse: keine Änderung

3.2.5 Video Encryption Algorithm (VEA)

Bei diesem Algorithmus wird der Datenstrom in zwei Teile aufgeteilt die miteinander XORed werden. Das Ergebnis und ein mit z.B. DES verschlüsselter Teilstrom ergibt das verschlüsselte Video. Kann natürlich auch auf Verschlüsselung von Teilen des Bitstreams in selektiven mehtoden angewendet werden.

- Sicherheit: sehr sicher
- Geschwindigkeit: doppelt so schnell wie naiv
- Datenstromgrösse: keine Änderung

3.2.6 Einsatzgebiete

Bei Video Verschlüsselung gibt es nicht nur ein Tradeoff bzgl. Sicherheit und Geschwindigkeit, es gibt da noch eine weitere Komponente: Das Einsatzgebiet.

Selective Algorithms kann man gut bei pay-per-view, video-on-demand und ähnlichen Anwendungen einsetzen, also überall wo SSMC (Single Server Multiple Client) im Spiel sind. Dabei kann man über 90% unverschlüsselt übertragen (P, B-frames), der Rest (I-frames) muss pro Zuseher verschlüsselt werden, weil entweder jeder Zuseher einen eigenen Schlüssel hat oder weil jeder Zuseher ein Video mit persönlichem Watermark erhält, um anonyme Raubkopien zu erschweren (insbesondere deren Verkauf).

Daher: Unverschlüsseltes multicasten, Verschlüsseltes singlecasten.

Dieses Einsatzgebiet muss nicht unbedingt hochsicher sein, die Video Durchsatzrate muss jedoch sehr hoch sein. Bei hunderten bis tausenden Clients dürfte diese Datenmenge bei Vollverschlüsselung kaum bewältigbar sein.

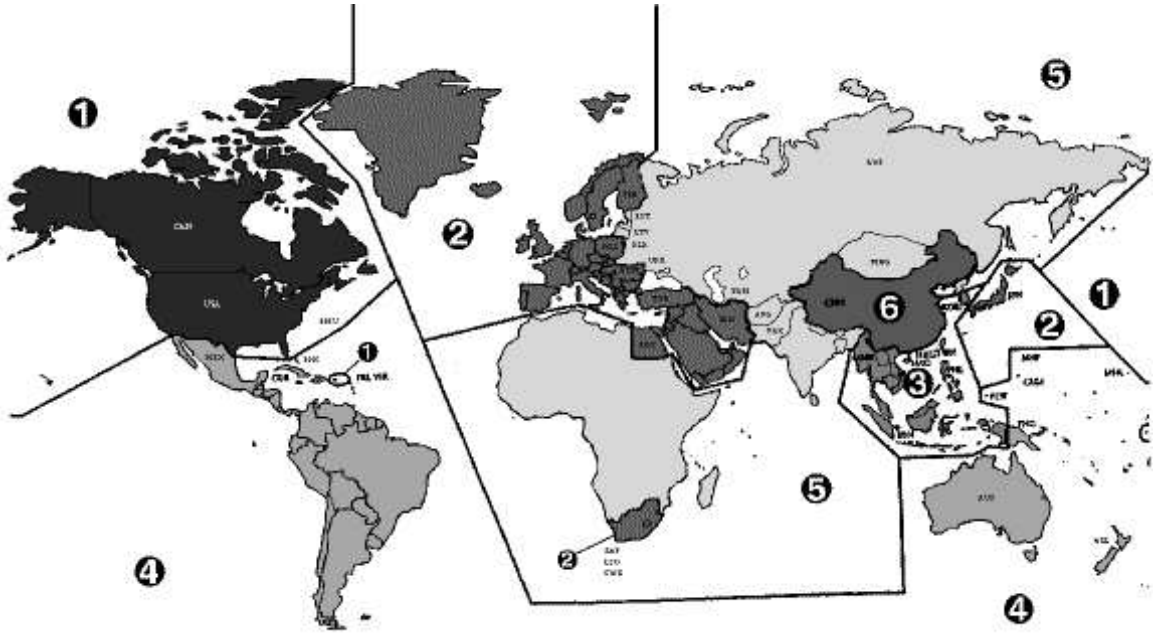
Anders bei Videokonferenzen und ähnlichem. Dort "berwiegt die Sicherheitsanforderung, Geschwindigkeit ist durch die wenigen beteiligten Personen eher kein Problem. Daher bietet sich eine Vollverschlüsselung mit einem der anderen Verfahren an.

3.2.7 DVD

Digital Versatile Disc oder auch Digital Video Disc. Hat maximale Speicherkapazität von 17 GB bei doppelseitiger 2 Layer Ausführung. Diese gewaltige Speicherkapazität bietet sich natürlich für hochqualitative Videspeicherung aber auch für nicht-komprimierte multiple channel Audiospeicherung an. Klarerweise stellt sich bei hochqualitativen digitalen Multimediadaten verstärkt die Frage noch Urherberrechtsschutz und Kopierschutzmechanismen.

Regional Codes

DVD-Video ist international nicht beliebig austauschbar. Auf Druck der amerikanischen Filmindustrie wurde die DVD-Welt im Februar 1997 in 6 verschiedene Regionen eingeteilt. DVDs können (müssen aber nicht, das steht dem Produzenten frei) einen Code enthalten, der zur Folge hat, dass die DVD nur abgespielt werden kann, wenn sie und das Abspielgerät denselben Regional-Code tragen. Das gibt den Hollywood-Studios die Kontrolle darüber in die Hand, wann und in welcher Version (Schnitt, Ton, Untertitel) ein Film auf den Markt kommt, wo zuerst und wo erst später. Europa hat den Regional-Code 2, zusammen mit Japan und Südafrika. Eine in den USA gekaufte und für den USA-Markt bestimmte DVD (Mit Regional-Code 1) lässt sich demnach auf einem in Europa gekauften DVD-Player nicht abspielen.



Regional Codes sind allerdings keinerlei Verschlüsselung, sondern sind nur ein einziges Byte auf der DVD-Disc, das vom Player überprüft wird. Bei einigen DVD-Playern kann der Regional Code vom Player verändert werden, dies ist allerdings nur einige Male möglich.

APS (Macrovision)

Dieser analoge Kopierschutz kann vom Produzenten auf der DVD-Disc durch "trigger bits" für einzelne Filmsequenzen gezielt ein- und ausgeschaltet werden. Das Macrovision System nutzt die unterschiedliche Arbeitsweise der automatischen Aussteuerung von Fernsehern und Videorekordern aus. Fernseher reagieren relativ träge auf Veränderungen des Eingangssignals während Videorekorder sehr schnell reagieren. Das Macrovision System verändert nun das Videosignal so, dass ein Fernseher ein korrektes Bild anzeigt, ein Videorekorder hingegen kein sauberes Bild aufnehmen kann.

Serial Copy Generation Management System (CGMS)

Man will aber natürlich auch verhindern, dass DVDs auf digitalem Wege kopiert werden, zum Beispiel auf die Harddisk (oder auf die DVD-R oder DVD-RAM) eines Computers. Es gibt daher im vom DVD-Player ausgegebenen Video-Signal auch ein "Serial Copy Generation Management System" (CGMS), das Kopien oder Kopien von Kopien verhindert.

In den Video-Daten werden die CGMS-Informationen (ob gar nicht kopiert werden darf, nur eine Generation oder beliebig) eingebettet:

- copy never (überhaupt kein Kopieren)
- no more copies (kein weiteres Kopieren mehr, dies ist bereits eine Kopie)

- copy one generation (diese Disk darf eine Generation kopiert werden)
- copy freely

Damit das System aber überhaupt funktioniert, muss sämtliche Hardware diese Informationen respektieren !

Content Scrambling System (CSS)

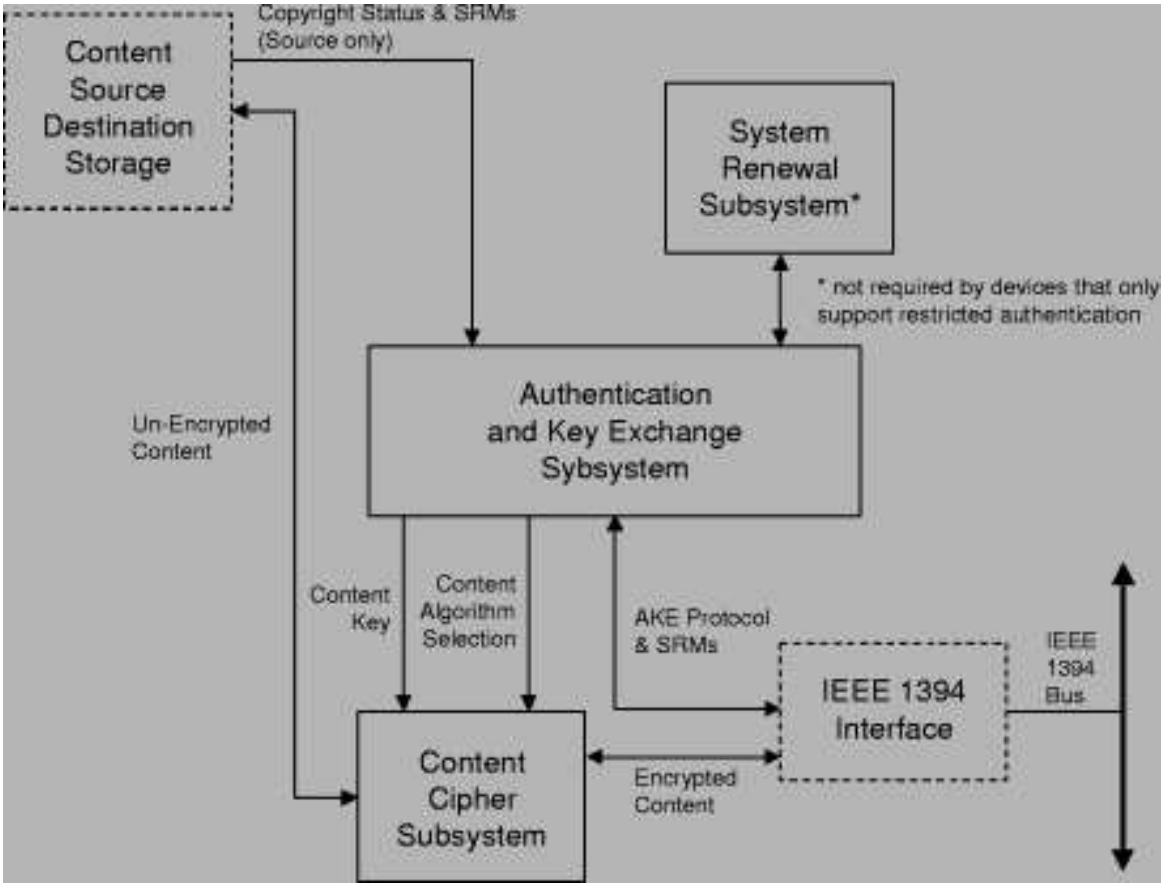
Zusätzlich werden die Daten auf der DVD durch CSS verschlüsselt und können dadurch nur auf DVD-zugelassenen Geäten abgespielt (i.e. entschlüsselt werden). Erst für eine Wiedergabe wird die Entschlüsselung durchgeführt (eine digitale Übertragung der entschlässelten Daten ist nicht zulässig), nach einem Code, den die Gerätehersteller beim Lizenzgeber beziehen müssen. Wie auch bei den vorherigen Verfahren, ist das System nur wirksam, wenn sich alle Gerätehersteller an die Spezifikationen halten, denn auch dieses Verfahren kann keinen, der die entsprechende Hardware hat, daran hindern, die verschlüsselte DVD-Disc 1:1 zu kopieren. Es hindert also den Normal-User eine Kopie zu erzeugen, nicht jedoch professionelle Kopierer.

Digital Transmission Content Protection (DTCP)

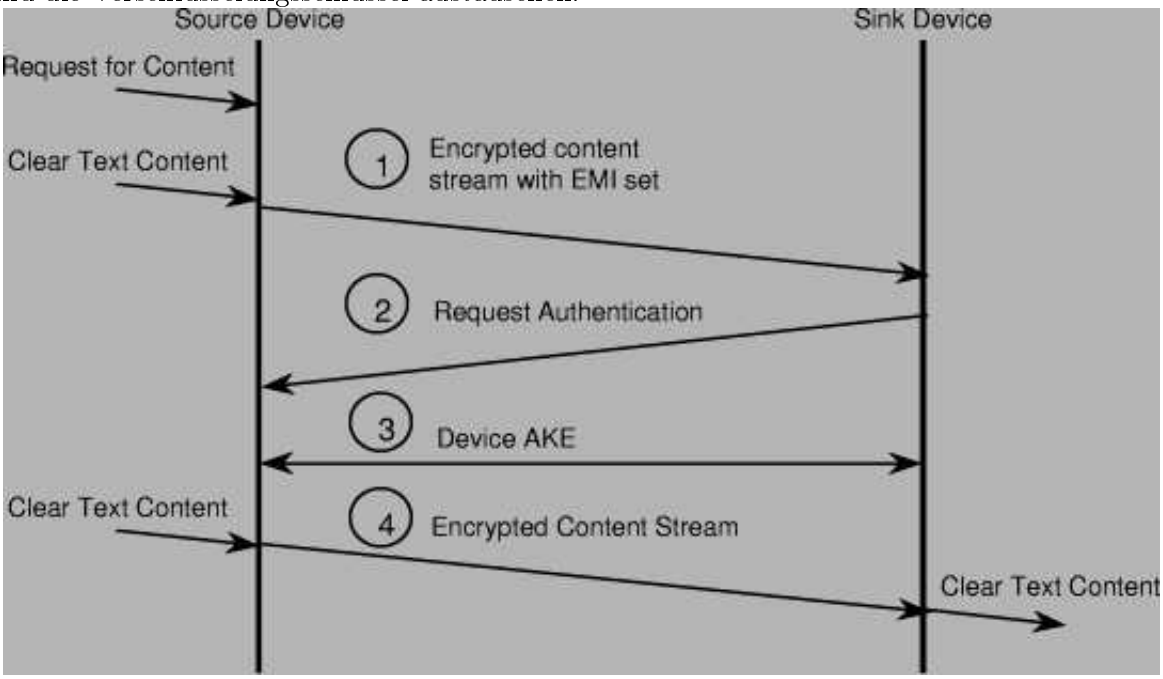
Um gegen Protokoll-Attacken bei der Kommunikation zwischen wei digitalen Geräten gefeit zu sein, wurde ein weiterer Kopierschutz erfunden. Geräte, wie DVD Player und digitaler Fernseher, die digital miteinander verbunden sind, authentifizieren sich gegenseitig, tauschen Schlüssel aus und bauen einen Kanal auf, über den das Video-Signal verschlüsselt übertragen wird. Ein digitaler Fernseher kann alle Daten empfangen und anzeigen, ein digitaler Videorekorder hingegen, kann nur Daten aufzeichnen, die dafür freigegeben sind (siehe CGMS). DTCP verwendet “deftige” kryptographische Algorithmen !

Als zusätzliche Sicherheit bietet das System noch Updates in Form von System Renewal Messages, die darüber informieren, welche Geräte nicht (mehr) standardkonform arbeiten. System Renewal Messages gelangen entweder über neuere DVD-Disks oder auch über digitales Fernsehen in das System und werden zwischen den über DTCP kommunizierenden Geräten ausgetauscht.

Wie dem Diagramm zu entnehmen, ist DTCP hauptsächlich in Verbindung mit IEEE 1394 (Firewire: ein digitaler Bus mit einer Übertragungsgeschwindigkeit von 100 - 400 MBit/s, der hauptsächlich für Multimedia-Daten ausgelegt ist) gedacht.



Bevor eine Übertragung von verschlüsselten Daten stattfinden darf, müssen sich die Geräte authentifizieren und die Verschlüsselungsschlüssel austauschen.



DIVX (Digital Video Express)

DIVX wurde von Circuit City eingeführt, es ist ein modifiziertes DVD-Format, das vor allem für den Videoverleih gedacht ist. Zusätzlich zu normalen DVD-Playern enthalten DIVX-Player noch eine Entschlüsselungsschaltung und ein Modem, um mit dem DIVX-Host zu kommunizieren. DIVX-Disc werden für ungefähr 5 Dollar gekauft werden können und danach 2 ganze Tage lang abgespielt werden können. Danach kostet ein weiteres Mal abspielen der Disc 3-4 Dollar, die Disc kann aber auch für ca. 15 Dollar ganz gekauft werden. In normalen DVD-Playern können DIVX-Discs gar nicht abgespielt werden.

Jede Divx-Disc hat eine zusätzliche Serien-Nummer, die der Divx-Player beim ersten Mal ausliest und sich merkt. In regelmässigen Abständen ruft der Divx-Player beim Divx-Host an, um Abrechnungsinformationen auszutauschen. Für die Verschlüsselung der Video-Daten verwendet Divx unter anderem Triple DES.

Der DVD-Hack

"We found that one of the companies had not encrypted their CSS decryption code, which made it very easy for us". Jeder DVD Player hat seinen eigenen Unlock-Schlüssel (sollte verschlüsselt sein !) und jede DVD Disc hat 400 verschieden verschlüsselte 5-byte CSS Schlüssel. So kann jeder Schlüssel von lizensierten Playern zuerst den CSS Schlüssel dekodieren und dann die Disc lesen. Im XingDVD Software Player (RealNetworks) war der der Schlüssel für CSS nicht verschlüsselt (nur die Software war "verwischt" - das ist aber ein grundsätzliches Problem !!!) und konnte einfach herausgelesen werden. Bei Softwarelösungen muss der Schlüssel immer irgendwo im Computer in unverschlüsselter Form liegen und kann mit entsprechendem Wissen herausgelesen werden. Diese Entwicklung könnte die Einführung/Verbreitung von schreibbaren DVD wesentlich verzögern. Reaktion des DVD Forums (Japan): "The circulation through the Internet of the illegal and inappropriate software is against the stream of copyright protection. Toshiba, which has led the establishment of the DVD format and is the chair-company of the DVD Forum, feels it is a great pity. In the future, the laboratories will be more actively conducting strict surveillance and take counter measures against illegal, inappropriate software and hardware in the market. Moreover, we believe that, based on the recent legislation, legal measures and steps will be taken by copyright holders against such violation of intellectual properties."

Kapitel 4

Multimedia Datenbanken

4.1 Video on Demand (VOD)

Video on Demand bezeichnet user-initiierte, single-gecastete Videoübertragung über ein Netzwerk mit voller VCR Funktionalität: FFD, REW, PAUSE, ZEITLUPE (im Gegensatz zu multicast oder broadcast). In diesem Bereich gibt es eine Vielzahl ähnlicher und nicht sauber definierter Begriffe, z.B.:

- VOD
- MOD: Movie on Demand - wie VOD nur ohne VCR Funktionalität
- ITV: Interactive TV - wie HDTV nur mit VCR Funktionalität und Web-Anbindung, e-commerce, ..

Offensichtlich sind die Anforderungen an Hard- und Software bei einer VOD Anwendung sehr hoch, insbesondere in den Bereichen Netzwerktechnologie und Videodatenbank bzw. Videoserver:

- große Speicherkapazität
- große Übertragungsbandbreite
- Echtzeitanforderung

Als Beispielmaterial können MPEG-2 Filme mit einer Datenrate von 3 Megabit/sec. (30 frames/sec.) dienen, die bei einer Filmlänge von 100 Minuten einen Speicherbedarf von 2.2 Gigabyte ergeben. Es gibt praktisch keine Studien die Aussagen über die tatsächliche Verwendung der verschiedenen VCR Funtionalitäten liefern, wohl aber Modelle für das Film- Auswahlverhalten (das offensichtlich nicht unwesentlich für das Design eines entsprechenden Videoservers ist). Meist wird für die Modellierung das Gesetz von Zipf verwendet: die Wahrscheinlichkeit den n-populärsten Film von M Filmen auszuwählen, beträgt C/n wobei $C = 1/(1+1/2+1/3+1/4+\cdots+1/M)$. Lokalität im Access-pattern ist von großer Bedeutung für Fragen wie Caching oder hierarchische Speichermodelle ! Zusätzliche Anforderung kann sein, dass verschiedene Clients bedient werden müssen, die unterschiedliche Anforderungen an das Filmmaterial stellen (insbesondere verschiedene Qualität bzw. Auflösung). Je nach Art der Kodierung des Filmmaterials (intrinsisch skalierbar oder skalierbar über enhancement layers) gibt es dann verschiedene Access-patterns je nach Struktur des momentanen Client-pool.

4.1.1 Die Single Server Architektur

Die meisten heute verwendeten VOD Systeme verwenden einen einzigen Server, je nach Größe des Systems von PCs bis hin zu massiv parallelen Supercomputern. Dieser Ansatz hab aber klare Einschränkungen:

- Skalierbarkeit: wenn die Anzahl der Clients steigt, ist die Kapazität einer Servers irgendwann erschöpft. Es gibt drei Ansätze dieses Problem zu meistern:
  - Replikation: Daten werden dupliziert und auf einem zusätzlichen Server abgelegt. Um die offensichtlichen Probleme im Bereich Load Balancing und wechselnde Access-pattern zu lösen wurden Replikationsalgorithmen vorgeschlagen, die Videopopularität und/oder Server Heterogenität berücksichtigen. Replikation führt aber in jedem Fall zu Resourcenverschwendung !
  - Partitionierung: die Videos werden in Gruppen aufgeteilt und jede Gruppe hat ihren eigenen Server. offensichtlich ist das Load-balancing problematisch, weil sich die Access-pattern zeitlich verändern. manche Server werden überlastet während andere unterbeschäftigt sind.
  - Multicasting: gleiche user-requests werden zusammengefaßt und durch Multicasting bedient. Nachteil ist das offenbare Fehlen von VCR Funktionalität (neuere Arbeiten lösen dieses Problem teilweise) und die Problematik der zeitlichen Koordination (nur bei großen Systemen mit viel usern und wenig Filmen denkbar !).
- Fehlertoleranz: Ein Serverausfall in diesem Modell führt zum Stillstand des VOD Systems. Replikation kann dieses Problem lösen, jedoch wird der Speicherbedarf wesentlich höher. Auch das Partitionieren löst dieses Problem nur zum Teil (die Filme des ausgefallenen Servers bleiben unerhältlich).

Die Lösung für diese Probleme sind parallele oder verteilte Video Server, wo die Video Daten über mehrere Server gestriped werden, wodurch Skalierbarkeit und automatisches Load-balancing garantiert wird sowie Fehlertoleranz auf Server Ebene ermöglicht wird (Annahme ist immer daß ein Video über alle Server gestriped wird).

4.1.2 Die Parallele/Verteilte Server Architektur

Da ein Client (also ein Video Dekoder) natürlich einen einheitlichen Videostrom erwartet, müssen die Ströme von unterschiedlichen Servern (durch das Striping entstanden) wieder vereinigt werden. Diese Aufgabe wird vom sg. Proxy Modul ausgeführt. Zusätzlich kann ein Proxy Caching Aufgaben durchführen und Daten Redundanzen benützen um Server Ausfälle zu maskieren. Ein Proxy ist ein Hardware oder Softwaremodul das die Systemkonfiguration kennt, insbesondere Anzahl und Adressen der Server N_s, Daten Lokation und Striping Policy. Es gibt drei Möglichkeiten, wo ein Proxy realisiert werden kann:

- Proxy at Server: Ein Proxy ist auf jedem Serverrechner realisiert. Meist gibt es mehr Clients als Server, d.h. jeder Proxy muß mehrere Clients bedienen. Die Server sind durch ein Netzwerk verbunden, Proxys kombinieren die Daten vom lokalen Server und anderen Servern in einen Videostrom und übertragen diesen zu ihren Clients. Ein Vorteil dieses Ansatzes ist, daß die Systemkonfiguration für die Clients verborgen bleibt. Nachteil ist der Verarbeitungs und Übertragungsoverhead: Daten müssen vom lokalen Server gelesen werden, zum richtigen Proxy übertragen, dort verarbeitet und dann zum Cleint übertragen werden.

- Independent Proxy: Die Proxys laufen auf einem oder mehreren eigenen Rechnern. Proxys und Server sind durch ein Netzwerk verbunden, jeder Proxy ist über Netzwerk mit mehreren Clients verbunden. Ebenso wie im ersten Ansatz wird die Systemkonfiguration vor den Clients verborgen. Außerdem gibt es keine Beeinflußung zwischen Proxy und Server Modul, was deren Implementierung vereinfachen kann. Nachteil ist, daß es noch mehr Verarbeitungs und Übertragungsoverhead gibt: alle Server schicken Daten zum Proxy, wo sie kombiniert und zu den Clients übertragen werden. Natürlich werden auch mehr Hardwareresourcen und Netzwerklinks benötigt.
- Proxy at Client: Auf jedem Clientrechner ist ein Proxy realisiert. Hier übertragen die Server die Daten direkt an die Clients bzw. ihre Proxies wo sie kombiniert und weiterverarbeitet werden. Es ist also deutlich weniger Datentransfer notwendig als in den beiden anderen Ansätzen. Außerdem gibt es keinen zusätzlichen Hardwarebedarf. Andererseits kann die Systemkonfiguration für die Clients nicht transparent sein.

Wenn der Computer auf dem ein Proxy läuft ausfällt, betrifft das im Proxy at Client Schema nur einen Client, in den anderen Schemen mehrere Clients. Die meisten Systeme verwenden das Proxy at Client Schema.

Werden die Videos über alle Server gestriped, so nennt man das wide server striping, wird nur über einen teil der Server gestriped heißt das short server striping. Meist wird wide striping verwendet. Es gibt zwei Methoden, wie Videos gestriped werden können:

- Time striping: Hier wird das Video als eine Reihe von Frames betrachtet. Striping wird der Framestruktur folgend durchgeführt. Die Anzahl L Frames pro Stripe Unit kann auch < 1 sein, dann spricht man von Subframe Striping. Beim Subframe Striping wird jeder Frame in k gleich große Teile geteilt und über die Server verteilt.
- Space striping: Hier wird das Video als ein Bitstream betrachtet. Das Video wird in gleich große (bezüglich bits) Stripe Units aufgeteilt was Speicher und Buffermanagement im Server wesentlich erleichtert. Das Striping wird unabhängig von Framegrenzen und Kodierungformaten durchgeführt.

Im Zusammenhang mit Striping müssen weitere Punkte beachtet werden:

- Anordnung der Stripes: Klassischer Zugang ist eine round-robin Verteilung der Stripe Units. Ein kleines Problem ist, daß Server i wahrscheinlich mehr Units speichert als Server j wenn $i < j$ weil die Video Länge i.A. kein Vielfaches der Stripe Size ist. Das kann ausgeglichen werden indem bei verschiedenen Videos an verschiedenen Servern mit dem Striping begonnen wird. Auch Randomisierung wird diskutiert, wo die Anordnung der Units innerhalb eines Stripes zufällig permutiert wird. Das hilft einerseits den Konvoi-Effekt zu begrenzen (d.h. ist ein Server überlastet, so wird die Überlastung durch identisches round-robin auf den nächsten Server üertragen, u.s.w.), andererseits ist restriping beim Hinzufügen neuer Disks weniger aufwendig (dafür ist der Durchsatz geringer).
- Load Balancing: beim Time striping hängt die Datenmenge die von jedem Server retrieved wird vom Frametyp ab - insbesondere ist das bei Interframe Kompression wichtig (I,P,B Frames in abnehmender Größe) - daher kann es bei Servern die öfter I-frames retrieven leichter zu Überlastung kommen. Besonders negativ wirkt sich aus, wenn N_s ein Vielfaches der GOP-Größe ist, da die identische ungleiche Lastverteilung durch das ganze Video hindurch erhalten bleibt. Dieses Problem kann gemindert werden, indem L als ganzzahliges Vielfaches der GOP-Größe gewählt wird, jeder Server also zumindest

eine ganze GOP pro Stripe unit retrieved. In der Praxis ist das aber nicht so einfach weil die GOP-Größe auch bei fix-GOP MPEG leicht variiert und außerdem manche MPEG Encoder variable GOP Größe haben, um die Qualität zu erhöhen. Subframe Striping löst zwar die Load-balance Probleme, aber die Verarbeitungskomplexität im Proxy steigt mit der Anzahl der eingesetzten Server (für jeden Frame müssen Subframe Daten von jedem Server kombiniert werden !). Beim Space-striping ist die Situation anders - zwar ist die Datenmenge für alle Server gleich (unit size X0 - X00 Kilobytes), die Vorraussetzung daß auch das Abspielen in konstanter Zeit passiert stimmt aber meist nicht (i.A. wird eine verschiedene Anzahl von Frames und Frameteilen in einem Unit enthalten sein, in constant-quality Video ändert sich ja auch die Video Bitrate). Hier kann Abhilfe über einen ausreichenden Client-buffer geschaffen werden.

Ein weiteres Load-balancing Problem kann durch FFD und REW ausgelöst werden. Es gibt zwei Ansätze wie FFD realisiert werden kann: einerseits wird ein eigener stream gespeichert, andererseits werden Frames weggelassen um die schnellere Abspielrate zu erreichen. Im ersteren Fall wird extra Speicherplatz benötigt, sonst treten keine Probleme auf. Der zweite Fall kann zu Load-balancing Problemen führen, v.a. beim Time-striping. Im einfachsten Fall ist die Stripe Unit 1 Frame, bei 4 Servern und doppelter Geschwindigkeit werden nur 2 Server aktiv sein ! Um solchen Effekten zu entgehen, müssen eigene Daten layout Schemen für VCR Operationen entwickelt werden.

- Redundanz: wie bereits erwähnt eröffnen parallele Video Server die Möglichkeit zu Fehlertoleranz auf Server Level - idealerweise sollte das System jede aktive Playback Session aufrechterhalten können wenn ein oder mehrere Server ausfallen. Neben Caching im Proxy kann v.a. serverseitige Redundanz verwendet werden um Ausfallssicherheit zu erreichen. Grundsätzlich kann Redundanz durch einfaches Kopieren (Mirroring) oder durch Einführung von Paritätsdaten erreicht werden. Im VOD Bereich ist grundsätzlich letzeres vorzuziehen, da die großen Datenmengen Mirroring teuer machen und im Serverausfall stetiges Weiterabspielen keinesfalls realisiert werden kann. Für Paritätsdaten ist es notwendig, daß die Stripe Units innerhalb eines Stripes identische Größe haben, was Time-striping für diese Anwendung ungeeignet macht ! Viele der denkbaren Varianten sind im RAID (Redundant Array of Inexpensive Disks) System realisiert.

4.1.3 Exkurs: RAID und Speichermedien für VOD

- RAID 0: Striping - keine Ausfallssicherheit !
- RAID 1: Striping und Mirroring - teuer, nur 50% Speicherauslastung !
- RAID 2-4: Striping mit einer oder mehrerer Parity Disks. Das j-te Bit der Parity Platte ergibt sich als XOR der j-ten Bits der restlichen Platten (RAID 4). Bei Ausfall einer Platte kann durch die parity Information diese Platte rekonstruiert werden. Problem: nach jedem Schreiben muß auch Parity Platte geändert werden. Bei Ausfall der Parity Platte langsames Recovery !
- RAID 5: Striping von Daten und Parity Information - das Bottleneck von RAID 4 fällt weg.
- RAID 6: wie RAID 5 nur mit höherer Redundanz.

Grundsätzlich kommen für VOD natürlich nicht nur klassiche Disks als Speichermedium in Frage - durch die hohe Lokalität der Zugriffe bieten sich hierarchische Lösungen an, z.B. RAM – RAID oder Disk Farms –

Tapes oder Optical Disks. Auf diese Art kann natürlich auch Ausfallssicherheit gewährleistet werden, wenn auch nicht in Echtzeit.

Eine weitere Frage im Zusammenhang mit VOD ist die nach optimalem Scheduling von Retrieval Requests auf einer Platte.

- First in - first out: durch Suchen zwischen dem innersten und äußersten Track geht für Multimedia Applikationen zuviel Zeit verloren.
- SCAN: während sich die Leseeinheit über die Platte bewegt, werden requests die "auf dem Weg liegen" bedient. Durch das out-of-order Schema werden allerdings Pufferbereiche benötigt.
- GSS (grouped sweeping scheme): hier wird versucht, den besten Kompromiss zwischen beiden vorigen zu finden: es werden Gruppen von Requests gebildet, innerhalb dieser Gruppen kommt SCAN zum Einsatz, die Gruppen werden aber in fixer Reihenfolge abgearbeitet. Je nach Systembeschaffenheit kann die Gruppenanzahl variiert werden (und man geht eher in Richting SCAN oder First in - first out).
- Earliest deadline first: speziell für zeitkritische Umgebungen: sind die Requests mit Deadlines versehen, gibt das die Reihenfolge der Abarbeitung an.

4.1.4 Fehlertoleranz

Im Bereich Fehlertoleranz bei der Übertragung gibt es zwei grundverschiedene Modelle. Forward Error Correction (FEC) sendet immer redundante Daten zum Empfänger. Dadurch muß ein Empfänger einen Server Ausfall nicht feststellen bzw. muß nicht festgestellt wer den welcher Server ausgefallen ist. Natürlich ist der Overhead groß. Eine Idee ist hier das progrssive Übertragen von redundanten Daten wenn ein hoher Redundanzgrad verwendet wird. Beim Übertragungsbeginn wird ein geringer Redundanzgrad verwendet der im Fall einer Serverausfalls durch einen Clientrequest erhöht wird. On-demand Correction (ODC) ist der zweite Weg bei dem nur redundante Daten bei einem Serverausfall gesendet werden. Das schwierige dabei ist eine Methode zu entwickeln, die Server Ausfälle schnell und sicher feststellt, zu viele falsche Alarmauslösungen führen wieder zu erheblichem Overhead !

4.1.5 Beispiele für kommerzielle parallele Multimedia Filesysteme

- Tiger Shark (IBM): für RS/6000 und SP2/3 Systeme, auf der SP2 bis zu 512 Prozessoren. Wide Striping wird verwendet, für Fehlertoleranz kann RAID (teuer und Bottleneck) oder ein selbstgezimmerter Replikationsmechanismus auf Blockbasis verwendet werden. Beim I/O wird earliest deadline first verwendet, wobei die deadline berechnet wird. Schon 1996 in Tokyo mit Support von 100 streams eingesetzt (SP2 mit 12 Prozessoren).
- XFS (SGI): für Challenge XL und Origin 2000 Systeme, verwendet B-tree Lookup (O(log n) bei n files) anstelle von extendible Hashing (O(1)) wie bei Tiger Shark. Die Datenreplikation passiert auf Plattenlevel, dadurch muß bei einem Ausfall die Mirrorplatte die gesamte Last übernehmen.
- Tiger (MS): verwendet ATM multipoint-to-point Verbindung zwischen den Server Knoten und einer Set-top Box. ADSL unterstützt multipoint-to-point aber nicht.

4.1.6 Transport Protokolle

VOD Systeme haben zwei grundlegen verschiedene Varianten um Daten vom Server zu requesten und zu transortieren. Die meisten Systeme lassen den Server Daten in kontrollierter Rate an die Clients liefern. Der Client puffert und rekonstruiert die Daten. Der Server fährt mit der Übertragung fort bis der Client ein Signal zur Beendigung sendet. Dieser Modus heißt Server-Push. Im Gegensatz dazu wird im Client-Pull Modus für bestimmte Video Daten Teile ein Request vom Client gesandt - der Server reagiert immer nur auf Requests. Die meisten single-Server VOD Systeme verwenden das Server-Push Modell. Für parallele VOD Systeme ist dieser Ansatz etwas komplizierter durch parallele Übertragung von unabhängigen Servern. Nach Senden des Requests durch einen Client starten alle Server die Session - da die Stripe Units i.A. nicht in der richtigen Reihenfolge eintreffen muß der Proxy rel. viel puffern. Im Fall von durch Netzwerkbeschaffenheit verschiedenen Delays wird kompensierende delay vorgeschlagen, ebenso gibt es Vorschläge für dedizierte Hochgeschwindigkeitsnetzwerke für die Server um perfekte Synchronisation zu erreichen. Dieser Ansatz heißt Concurrent-Push, und wurde durch rel. komplizierte Weiterentwicklungen (im Striping und Disk-request scheduling) für Systeme bis 10000 Nutzern nutzbar gemacht. Das Client-pull Modell läßt sich viel einfacher auf parallele Server übertragen, da ein Client (bzw. der Proxy) einen expliziten Request an einen Server nach bestimmten Daten schickt. Dadurch ist die Synchronisierung implizit.

Bei single-Server VOD Systemen (auf die wir uns im folgenden beschränken) gibt es verschiedene Aspekte bezüglich der Optimilität des einen oder anderen Transport Protokolls. Im Fall der Request Architektur (client pull) wird zwar nur eine geringe Bandbreite zwischen client und server benötigt, die Wartezeit kann minimiert werden, jedoch sind bei diesem Ansatz nur eine kleine Anzahl Clients bedienbar (server bottleneck). Andererseits benötigt die Broadcast Architektur (server push) höhere Bandbreite und eine höhere Anzahl von logischen Kanälen (siehe später), hat offensichtlich Probleme mit der Wartezeit der clients, dafür kann aber eine wesentlich höhere Anzahl von Clients bedient werden. Im Folgenden werden einige Vertreter und auch Modifikationen dieser klassischen Protokolle besprochen. Die wichtigsten (natürlich einander z.T. widersprechenden) Kriterien/Ziele beim Design eines solchen Protokollles sind:

- minimale client Wartezeit auf den Filmbeginn
- minimale Bandbreite
- minimale Anzahl von Kanälen
- geringer Speicherbedarf der Set-top Box (STB)

Zusätzlich kann man sich überlegen unterschiedliche Protokolle für “cold” (eher client pull) und “hot” (eher server push) video objects zu verwenden. Um die vorher erwähnten Probleme zu lösen wurden eine Vielzahl von Protokollen entwickelt die einerseits die Broadcast Architektur verbessern, andererseits beide Architekturen kombinieren.

- Verbessertes Broadcast
  - Staggered Broadcast: in fixen Abständen wird jedes Video gebroadcasted (z.B. für hot videos die Abstände kleiner machen). Wartezeit entspricht genau dem Abstand, Bandbreite wächst natürlich mit dem Verkleinern der Abstände.

  - Batching: serverseitig wird abgewartet, bis mehrere requests für ein Video eingelangt ist bevor es gemulticastet wird. Nimmt auf "cold" und "hot" video objects Rücksicht, für einzelne Benutzer kann aber keine minimale Wartezeit garantiert werden.
  - Pyramid Broadcast (auch Skyscraper Broadcast): die zur Verfügung stehende Bandbreite B wird in K logische Kanäle gleicher Bandbreite geteilt. Die Videos werden in K Segmente steigender Grße aufgeteilt, jedes Segement wird pausenlos auf seinem Kanal gesendet. Ist das Verhältnis zwischen zwei Segmenten ganzzahlig (es ist in jedem Fall konstant), spricht man von synchronem Pyramid Broadcast. Ein Client wechselt nach jedem Segment den Kanal. Die Downloadrate wird als grßer als die Abspielrate vorrausgesetzt.
  - Harmonic Broadcast: Das Video wird in n gleich lange Segmente aufgeteilt. Jedes dieser Segmente wird wiederholt über einen Kanal der Bandbreite b/i gebroadcastet, wobei b die Abspielrate ist.
  - Cautios Harmonic Broadcast: Ähnlich wie vorher, nur wird auf Kanal 2 abwechselnd das zweite und dritte Videosegment ebenfalls mit Abspielrate gebroadcastet. Weiters wird auf Kanal i Segement i+1 mir Bandbreite b/i gesendet.
  - Polyharmonic Broadcast: Mit dem Abspielen wird erst begonnen, wenn aus allen Kanlen von einem Harmonic Broadcast eine bestimmte Anzahl Segemente heruntergelanden wurden.

- Kombinierte Protokolle
  - Piggbacking: die Playbackraten werden verändert um ab einem bestimmten Zeitpunkt nur einen Stream zu benötigen. Fehlende Daten wie bei dynamic Multicast geliefert.
  - Dynamic Multicast: der Multicast Tree ist dynamisch, d.h. neue requests werden aufgenommen und bedient. Die Daten werden in der STB zwischengespeichert. Der Server liefert duch einen *Patching*stream die fehlenden Filmanfangsdaten nach.
  - Transition Patching: Da beim Patching die Streams immer länger werden (gegen Filmende!) werden auch Patchingstream gepatched. Also rekursives Patching. Wird auch als Stream-merging bezeichnet, die Frage nach der optimalen Stategie wann welche streams gemerged werden ist nicht trivial.
  - Catching: hier werden Broadcast Protokolle mit Patching kombiniert, wieder hoher Anspruch an STB.
  - Selective Catching: über eine Optimierung wird definiert, wie oft Patching bzw. Rebroadcast durchgeführt werden.
  - Partial Preloading in STB: hier werden die Anfangsteile der meistgesendeten Videos immer auf der STB vorgespeichert. Kann mit verschiedenen Broadcastvarianten kombiniert werden (heißt z.B. in der Kombination mit Pyramid Broadcast Mayan Temple Broadcast)

Es mag vielleicht überraschen, daß auch in diesem Bereich formale Methoden gut eingesetzt werden können. Als Beispiel soll hier synchrones und asynchrones Pyramid Broadcasting dienen.

Serverseitig gibt es die folgenden Parameter:

B ist die Bandbreite des Übertragungsmediums

K Anzahl der logischen Kanäle

B/K Bandbreite pro Kanal

D Videogröße

s_i Videosegmente

D_i ist die Größe des Videosegments s_i, $D = D_1 + D_2 + \cdots + D_K$

$\alpha = D_{i+1}/D_i$

Clientparameter sind:

d Downloadrate, wobei $d = B/K$.

c Abspielerate

Clientseitig wird auf den Beginn des ersten Segments gewartet, dann wird mit Rate d heruntergeladen. Laden und Abspielen kann gleichzeitig geschehen, da $d > c$ vorrausgesetzt wird. Jeder der Kanäle benötigt eine Bandbreite zumindest so groß wie die Downloadrate. Nachdem ein Segment vollständig heruntergeladen wurde, beginnt der Client das nächste Videosegment auf dem nächsten Kanal herunterzuladen.

Die maximale Wartezeit beim asynchronen Pyramid Broadcast beträgt $t_{ap} = \frac{D_1}{d}$ da die Clients nur auf den Beginn des ersten Segments warten müssen. Will man diese Wartezeit in Abhängigkeit von der Anzahl der Kanäle ausdrücken, kann man folgendermaßen vorgehen:

$$D = \sum_{i=0}^{K-1} D_1 \alpha^i = D_1 \frac{\alpha^K - 1}{\alpha - 1} \ .$$

Daraus folgt $D_1 = D\frac{\alpha-1}{\alpha^K-1}$. Ersetzt man nun in obigem Ausdruck D_1, erhält man für $t_{ap} = D/d\frac{\alpha-1}{\alpha^K-1}$. Man sieht, daß die Client Wartezeit exponentiell abnimmt, wenn die Anzahl der Kanäle linear steigt.

Bei diesem Ansatz kann die Wartezeit für das $i+1$-te Segment so groß sein wie die Downloadzeit für dieses Segment (wenn gerade der Anfang verpaßt wurde). Um ein Betrachten des Videos ohne Unterbrechung zu garantieren, darf diese Zeit nicht größer sein als das Zeitintervall zwischen dem Ende des Herunterladens des i-ten Segments und dem Ende des Ansehens des i-ten Segments.

$$t_{download}(s_{i+1}) = t_{maxwait}(s_{i+1}) \leq t_{consume}(s_i) - t_{download}(s_i)$$

Das bedeutet, daß $\alpha\frac{D_i}{d} = \frac{D_{i+1}}{d} \leq \frac{D_i}{c} - \frac{D_i}{d}$. Daraus folgt, daß $\alpha \leq d/c - 1$. Im besten Fall nimmt α diesen Wert also an.

$$t_{ap} = D/d\frac{d/c - 2}{(d/c - 1)^K - 1} \ .$$

Beispiel: Sei $D = 21600Mbit$, $c = 3Mbit/sec$, $B = 45Mbit/sec$. Bestimmen Sie für $K = 2, \ldots, 7$ die Wartezeit für asynchrones Pyramid Braodcast und das entsprechende α und ermitteln Sie die optimale Anzahl der Kanäle. Wie sehen die Werte im Vergleich zu konventionellem Broadcast aus ?

Im Fall des synchronen Pyramid Broadcast sind die Vorraussetzungen identisch, nur daß α einen Integerwert haben muß und gefordert wird, daß das Broadcasting der verschiedenen Segmente synchronisiert ist. Dadurch kann beim Kanalwechsel der Beginn eines Segments nicht knapp verpaßt werden, weil der Beginn der Segmente auf verschiedenen Kanälen synchron ist. Diese Synchronisierung bewirkt, daß die Wartezeit um das $i+1$-te Segment laden zu können nicht größer ist als $\frac{\alpha-1}{\alpha}\frac{D_{i+1}}{d}$. Daraus folgt:

$$t_{maxwait}(s_{i+1}) = \frac{\alpha-1}{\alpha} t_{download}(s_{i+1}) \leq t_{consume}(s_i) - t_{download}(s_i)$$

Das bedeutet, daß $\frac{\alpha-1}{\alpha}\frac{D_{i+1}}{d} = (\alpha-1)\frac{D_i}{d} \leq \frac{D_i}{c} - \frac{D_i}{d}$. Daraus folgt, daß $\alpha = d/c$ (warum ?). Im besten Fall nimmt α diesen Wert also an.

$$t_{sp} = D/d\frac{d/c-1}{(d/c)^K - 1} \ .$$

Beispiel: Bei gleichem Setting wie in vorigem Beispiel sei nun $K = 5$. Wieviel bringt der synchrone Ansatz im Gegensatz zum asynchronen ? Welche Auswirkung auf die Wartezeit hat eine Verringerung von K auf $K = 4$ bei gleichbleibendem d ? Und im Vergleich zum asynchronen Fall mit $K = 4$ und $d = 11.25$?

4.1.7 Caching und Replacement von Cache Daten

Um in verteilten oder hierarchischen VoD Systemen geringe Wartezeiten realisieren zu können, müssen sog. Caching oder Prefetching Schemen verwendet werden. Wichtige Daten (in welchem Sinn auch immer) werden "nahe" am Client abgelegt, sodaß schneller Zugriff erfolgen kann. Dies kann z.B. im Proxy oder in der STB erfolgen (was beim Proxy-at-Client Modell beinahe identisch sein kann).

Klassische Caching Policies sind die folgenden:

- Object-level Caching: ganze Videos werden gecached, durch die große Datenmenge ist es nur möglich wenige Videos zu cachen. Dadurch werden die meisten Videos ohne die Hilfe des Proxys bedient.
- Prefix Caching: ein kleiner Anfangsteil von allen Videos wird gecached (siehe z.B. Mayan Temple Broadcast). Offensichtlicher Nachteil ist dass es unvernünftig ist gleich große Speicherbereiche für populäre und nicht-populäre Videos zu verwenden.
- Popularity-based partial Caching: In Abhängigkeit von der Popilarität des Films wird ein mehr oder weniger großer Teil des Videos gecached.

Ein weiteres Thema in diesem Zusammenhang ist wie man die Datenbelegung des Cache und dessen Austausch organisiert. Da der Cachebereich beschränkt ist muß es Verfahren geben die festlegen, welche Daten im Cache durch neue ersetzt werden. Auch hier gibt es zwei klassische Verfahren:

- LRU (Least Recently Used): die Daten deren letzte Nutzung am weitesten zurückliegt, werden ersetzt.
- LFU (Least Frequently Used): die Daten, die in einem bestimmten Zeitintervall am wenigsten requestet werden, werden ersetzt.

Klar ist, dass die Entscheidung welches das beste Verfahren ist ganz wesentlich vom Typ der Daten abhängt, d.h. man sollte die Zugriffsmuster auf die entsprechenden Daten kennen und optimalerweise vorhersagen können. Das erklärt auch die Wichtigkeit der Modellierung von Videopoularität und Zugriffswahrscheinlichkeiten.

Im Folgenden wird ein Schema zum Popularity-based partial Caching und ein entsprechendes Cache-Datenaustauschmodell vorgestellt. Im Zusammenhang mit Multimedia Daten ist es von Vorteil von "access degree" an Stelle von "access frequency" zu sprechen, da es mehr auf die tatsächlich übertragene Datenmenge ankommt als auf die Häufigkeit des Beginns einer Übertragung. Der Proxy mißt kontinuierlich die Datenmenge die für jedes Video von den Clients konsumiert wird. $DATA_{i,j}^k$ ist die Datenmenge des Videos i das vom Client j in der k-ten Beobachtungsperiode konsumiert wird. Sehen c_i Clients in der Periode k das Video i, ist die gesamte Datenmenge des Videos i in dieser Zeit $DATA_i^k = \sum_{j=1}^{c_i} DATA_{i,j}^k$. Der Proxy berechnet daraus den gewichteten Mittelwert der abgespielten Daten $D_i^k = \alpha DATA_i^k + (1-\alpha)D_i^{k-1}$ mit $0 < \alpha \leq 1$ und α ist der Gewichtungsfaktor für neue Information. Es sollte groß gesetzt werden für Daten die schnell altern (wie z.B. News on Demand NoD), jedoch klein für Daten die langsam altern (und lange gleichmäßig oft requestet werden, wie z.B. Filme mit PG-18).

Die Popularität P_i^k eines Videos i, $i, \ldots, N$ wird ausgedrückt durch: $P_i^k = \frac{D_i^k}{\sum_{m=1}^{N} D_m^k}$.

Popularity-based partial Caching berechnet nun periodisch die optimale Größe der zu cachenden Anfangsteile der Videos basierend auf der Popularität der Videos wie folgt. Zuerst wird die optimale Größe des Initialsegments des populärsten Videos berechnet, dann die des zweitpopulärsten und so weiter. Sei L_i die Größe des Videos i, $\hat{s}_i$ die optimale Größe des Anfangssegments und C die Gesamtgröße des Cache. Die Videos werden nach der Größe von D_i^k absteigend sortiert und die modifizierte Popularität $\hat{P}_i^k$ des Videos i unter den Videos $i, i+1, \ldots, N$ wird wie folgt berechnet: $\hat{P}_i^k = \frac{D_i^k}{\sum_{m=i}^{N} D_m^k}$. Die optimale Größe des Initialsegments jedes Videos wird dann berechnet als:

$$\hat{s}_i = \min(\hat{P}_i^k (C - \sum_{m=1}^{i-1} \hat{s}_m), L_i) \ .$$

Diese Werte werden periodisch berechnet und in Form einer Tabelle gespeichert und upgedated. Sobald ein Client einen request sendet, beginnt der Proxy sofort mit der Übertragung der daten im Cache und holt die restlichen Daten vom Server. Sobald ein bestimmtes Video requestet wird, führt der auch caching und replacement der Cachedaten durch. Sei s_i die Größe des Teiles eines Videos i der sich gegenwärtig im Cache befindet. Kommt der Clientrequest nach Video i zum Proxy und ist $s_i < \hat{s}_i$, so speichert der Proxy $\hat{s}_i - s_i$ zusätzliche Daten dieses Videos. Anderenfalls wird natürlich nicht mehr gespeichert. Ist $\hat{s}_i < s_i$ wird umgekehrt $s_i - \hat{s_i}$ an Daten entfernt.

Eine ähnliche Methode, die ebenfalls versucht Short-term Popularitätsmuster und Long-term Popularitätsmuster für ein Prefetching Schema einzusetzen ist LLBF (Life-cycle based Frequency). Hier wird auch die Entwicklung der Popularität verwendet. $LBF_k = f_k + r_k$ mit f_k die Zugriffsfrequenz im betrachteten Zeitfenster k und r_k die Fluktuation der Zugriffsfrequenz.

$$r_k = \alpha(f_k - f_{k-1}) + (1-\alpha)r_{k-1}$$

α ist das Gewicht der Fluktuation im aktuellen Beobachtungszeitraum. Es werden von allen Videos die

LBF_k Werte berechnet und in jedem Zeitfenster k wird eine bestimmte Anzahl R Videos (oder Videoteile) gecached, nämlich diejenigen mit den größten LBF_k Werten. Dieses Schema bevorzugt insbesondere Videos mit steigender Popularität und ersetzt erst solchem deren Popularität wieder im Sinken begriffen ist.

Aus diesen Schemata wird klar, dass es höchst wünschenswert ist, Modelle angeben zu können, wie sich Short-term Popularität und Long-term Popularität modellieren lassen ohne ständig Messungen durchführen zu müssen (es stellt sich z.B. heraus, dass Zipf's law zwar eine brauchbare Approximation ist, aber verbessert werden kann). Um das Langzeitverhalten von Videopopularität modellieren zu können, werden beispielsweise folgenden Daten verwendet:

- Berichte von VoD Systemen: kaum erhältlich, kaum Literatur !
- Kinobesuchszahlen: diese Werte modellieren nur das anfängliche Benutzerinteresse, weil nur die aller beliebtesten Filme lang genug im Kino bleiben. (− > Modell für Short-term Popularität)
- Filmverleihe: liefert Information wie rel. wenige Nutzer auf eine große Menge von Filmen zugreifen. Das Initialinteresse kann nicht danach modelliert werden, weil es nur eine begrenzte Anzahl von Filmkopien gibt. (− > Modell für Long-term Popularität)
- Filmmagazine: gute Durchschnittsdaten, viele Nutzer, viele Filme, regionale Unterschiede werden ausgeglichen.

Es gibt neben VoD weitere, relativ ähnliche Multimedia Services, die jedoch ganz unterschiedliche Kennparameter haben können:

- NoD: News on Demand - die einzelnen items haben eine sehr kurze "Lebensdauer", Interesse fällt extrem schnell, meist werden mehrere Artikel gelesen, d.h. die Popularitätsspitzen sind mehr verteilt.
- Video Präsentation: verwendet Teile von mehreren Videos, d.h. keine durchgängige Verwendung eines Datenstroms, an Kontrollpunkten kann es verschiedene Optionen geben.
- Video Browsing: extrem kurze Teile von Videos (oder Keyframes) werden benötigt, kaum regelmäßige Zugriffsmuster (ein DVL - Digital Video Library - Service).

4.2 Visual Information Search and Retrieval (VIS and VIR)

Die Aufgabenstellung in diesem Gebiet ist, aufgrund einer Abfrage aus einer Datenbank mit visuellen Daten ein Bild oder ein Video (oder nur Teile davon) zu retrieven. Diese Aufgabe ist sehr komplex, da allein schon verschiedene Arten von Daten im Zusammenhang mit Bildern oder Videos vorkommen:

- Inhalt-unabhängige Metadaten: Datenformat, Name des Autors, Datem der Aufnahme, Copyright, ...
- Inhalt-abhängige Metadaten: diese Daten haben mit Wahrnehmung zu tun, z.B. Farbe, Textur, Bewegung
- Inhalt-beschreibende Metadaten: semantischer Inhalt

Der Typ der Daten verwendet wird um visuelle Daten zu beschreiben hat natürlich Auswirkungen auf das VIS und VIR ! VIR Systeme der ersten Generation ermöglichen Zugang zu visuellen Daten durch Text-basierte Suche, z.B. "Bilder von Cezanne mit Landschaft. Alle obigen Datentypen werden durch Keywords und Skripts beschrieben. Retrieval erfolgt aus klassischen relationalen oder objektorientierten Datenbanken, existierende Abfragesprachen wie SQL können zur Query-Formulierung verwendet werden. Für viele Anwendungen sind solche Systeme aber nicht geeignet, da sich die meisten wahrnehmungsrelevanten Aspekte nicht effizient durch Texte/Keywords beschreiben lassen:

- Wahrnehmung von Form, Struktur und Bewegungseffekten in Videos lassen sich verbal kaum beschreiben
- Ähnlichkeit in Bezug auf Wahrnehmung ist schwierig in Worte zu fassen
- Textbeschreibungen von Inahlt-abhängigen und inhalt-beschreibenden Metadaten geben die Wahrnehmung des Schreibers/Textverfassers wieder, die mit denen des Systembenutzers nicht viel zu tun haben müssen.

Beispiele: Vererbung (Hund - Dackel), Relation (Mann beisst Hund - Hund beisst Mann)

Zusätzlich ist der Vorgang des Versehens einer visuellen Datenbank mit Keywords, die Annotation, ein extrem zeitaufwendiger Vorgang. Auch hier wurden semiautomatische Systeme entwickelt, die den Annotator durch vorgefertigte Icons unterstützen (Icon-Annotation). Ein weiterer Ansatz, Stratification oder Source Annotation, das Verwenden einer Daten Kamera, die ein Video zumindest sofort mit inhalt-unabhängigen Metadaten annotiert. Automatische (semantische) Annotation ist aber mindestens ebenso futuristisch wie bemannte Raumfahrt zu benachbarten Sonnensystemen - daher werden Methoden benötigt, die visuelle Inhalte beschreiben können und die mehr an menschlicher Wahrnehmung orientieren. VIR Systeme der zweiten Generation bieten neben Abfragemöglichkeiten auf Textebene auch solche auf Wahrnehmungsebene, wie objektive Maße von visuellen Inhalten oder geeignete Ähnlichkeitsmodelle, wie z.B. Pixelverteilungen, Farbcharakteristik, parametrisierte Formbeschreibungen u.v.m. Aus diesem Grund sind Bildverarbeitung, Mustererkennung und Computer Vision integrale Bestandteile eines solchen Systems.

Um nun Bilder entsprechend wahrnehmungsrelevanter Eigenschaften zu retrieven wird für jedes Bild in der Datenbank eine Menge von Unterscheidungsmerkmalen (Features) vorberechnet. Abfragen werden durch visuelle Beispiele durchgeführt. Um die Abfrage zu starten spezifiziert der Benutzer relevante Features, wichtige Parameterbereiche und ein Ähnlichkeitsmass. Die zur Abfrage verwendeten visuellen Beispiele können vom Benutzer erstellt werden (gezeichnet) oder aus anderen Bildern gewonnen werden:

- Ikon-Abfrage: vordefinierte Ikons werden ausgewählt und durch Boolsche Operatoren verknüpft. Das kann einer textuellen Abfrage am nächdsen kommen. Ikons können auch benutzt werden um örtliche Beziehungen zwischen Objekten in Bildern auszudrücken.
- Abfrage durch Malen: Farbflächen und ihre örtliche Anordnung können dargestellt werden. Erfolgt hängt auch von künstlerischen Qualitäten des Abfragers ab.
- Abfrage durch Sketch: eine Objektsilhuette wird gezeichnet. Gleiche Einschränkung wie vorher.
- Ein Prototypisches Bild wird für die Abfrage verwendet.

Das System berechnet dann die Ähnlichkeit zwischen dem visuellen Inhalt der Query und den Bildern in der Datenbank. Da perfekte Ergebnisse auf Anhieb kaum erwartet werden können (bzw. dem Benutzer nicht klar ist, was ein perfektes Ergebnis wäre) versucht das System vor allem die Anzahl der Misses (falsch negative Antworten) so gering wie möglich zu halten (bei gleichzeitigem in Kauf nehmen von mehr falsch positiven Antworten). Im Folgenden wird sog. Relevanz Feedback durchgeführt, d.h. der Benutzer kennzeichnet die besten Ergebnisse und started die Query von neuem. Dieses Ähnlichkeitsbasierte Retrieval unterscheidet sich durch zwei wesentliche Aspekte von Matching:

- Das Resultat von Matching ist binär, beim Ähnlichkeitsbasierten Retrieval wird eine Ordnung der Datenbankinhalte erstellt.
- Im Gegensatz zum Matching ist beim Ähnlichkeitsbasierten Retrieval der Benutzer in der Abfrageschleife, d.h. der Bedarf nach KI im System ist geringer.

Es ist klar, daß der Ergebnisvisualisierung eine zentrale Rolle zukommt: Stichwort ist hier Browsing ! Typischerweise geschieht eine Abfrage aus so einem System im Wechsel zwischen Browsing und Abfragespezifizierung.

Ein wichtiger Begriff im Zusammenhang mit VIS und VIR ist Image und Video Indexing: in der Literatur wird damit meist salopp der Vorgang der Feature-extraction **und** der anschließenden Indizierung (effiziente Verspeicherung für Suchzwecke) bezeichnet. Hier soll Indizierung im engeren Sinn verstanden werden. Wird die Anzahl von Bildern oder Videos in einer Datenbank zu groß wird der Aufwand für sequentielles Scannen zu groß. Index Strukturen müssen irrelevante Bilder/Videos ausscheiden ohne die gesamte Datenbank durchgehen zu müssen. Werden in einer Query Keywords und andere Strings verwendet, können hashing Tabellen oder Signatur files zum Indexing verwendet werden. Im Falle von visuellen Eigenschaften die als Punkte in einem mehrdimensionalen Feature-raum dargestellt sind können point access methods (PAMs) zur Indizierung verwendet werden (günstig sind dynamische Strukturen). Diese Methoden funktionieren für hochdimensionale Vektoren schlecht (über 10), daher muß man zuerst eine Projektion auf Räume mit niedererer Dimension durchgeführt werden.

Beispiele: Ausnutzen der Dreiecksungleichung, Gitter, K-d Bäume, R-Bäume, SS-Bäume

Im Videobereich sind folgende Schritte zusätzlich zum Imagebereich notwendig/möglich:

- Scene cut detection: image domain vs. compressed domain (siehe Praktikum), motion vectors
- Camera movement detection
- Key frame extraction: in Abhängigkeit von der Kamera- und/oder Objektbewegung ein oder mehrere I Frames; es können mehrere Frames zu einem Bild kombiniert werden (durch Überlagerung)
- Objekte können durch gemeinsame Bewegungsmuster automatisch erkannt werden

Klassische Maße um die Leistung eines VIR Systems zu messen sind Recall und Precision. Sind Bilder relevant im Bezug auf die Abfrage, so bezeichnen wir die Anzahl der ausgegebenen Bilder mit A (korrekt positiv), die Anzahl der nicht ausgegebnen mit C (inkorrekt negativ). Im nicht-relevanten Fall bezeichnen wir die Anzahl der ausgegebenen Bilder mit B (inkorrekt positiv), die Anzahl der nicht ausgegebnen mit D (korrekt negativ).

$$Recall = A/(A+C), Precision = A/(A+B).$$

Für Recall und precision werden mehr als 1000 Bilder benötigt um vernünftige Ergebnisse zu bekommen. Andere Metriken wären die durchschnittliche Anzahl an Feedbackrunden um ein gutes Ergebnis zu erzielen oder die Komplexität.

Kommerzielle Systeme:

- QBIC (IBM): das vollständigste System, für Bilder und Videos; unter IBM OS/2 und UNIX.
- Virage Search Engine: Bilder und Videos; für Informix und Oracle.
- Visual Retrievalware: nur Bilder; erhältlich für fast alle BS, Internetanbindung inkludiert.

Darüberhinaus gibt es eine Unmenge an experimentellen Systemen. Im Folgenden werden die wichtigsten Features für inhaltbezogenes Retrieval besprochen.

4.2.1 Retrieval durch Farbähnlichkeit

Beim Gebrauch von Farbe für Retrieval Anwendungen muss beachtet werden, daß es erhebliche interkulturelle Unterschiede in der sprachlichen Bedeutung Bedeutung von Farben gibt, auch wenn die Übersetzung gemeinhin als korrekt angesehen wird. Dieses problem kann durch "klickbare Farbtafeln leicht umgangen werden.

Folgende Fragestellungen sind denkbar:

- Suche Bild das spezifizierte Farbe(n) in einem bestimmten Verhältnis enthält (leicht - in Histogrammen die Pixel der Bars zählen)
- Suche Bild dessen Farben ähnlich sind wie die im angegebenen Beispiel: hier können Farbhistogramme eingesetzt werden, für die Differenzbildung können verschiedene Abstandsmaße verwendet werden. Das funktioniert nicht besonders gut. Deutlich besser ist die Verwendung von "Histogramm Intersection, wo im wesentlichen die Schnittmenge von zwei Histogrammen ermittelt wird. Eine Verbesserung ist inkrementelle Intersektion wo nur die größten Histogrammbins verwendet werden. Anstelle eines globalen Histogramms können auch mehrere lokale Histogramme verwendet werden und die vorigen Verfahren darauf angewendet werden. Farb-coherence Vektoren können verwendet werden um auch örtliche Information zu berücksichtigen. Dabei wird für jede Farbe die Anzahl der Pixel bestimmt, die in einem zusammenhängenden Gebiet gleicher Farbe sind, und die Anzahl der isolierten Pixel. Diese Vektoren können dann ebenfalls zur Distanzmessung verwendet werden.
- Suche Bild mit Farbregionen wie im gegebenen Beispiel: dies setzt eine Segmentierung in Farbregionen voraus (was bekannterweise nicht so einfach ist). Meist werden die Regionen stark simplifiziert bzw. mit multiresolution Darstellung gearbeitet. Viele Tools bieten eine Hilfe zur Segmentierung von Bildern durch einfaches iteratives Region Growing an.
- Suche Bild mit einem bekannten Objekt durch dessen Farbbeschreibung: hier bieten sich wieder Methoden mit lokalen Histogrammen an.

- Suche Bild das durch Farbe ähnliche Effekte erzeugt wie im gegebenen Beispiel oder der Beschreibung: für diesen Zweck können Erkenntnisse aus der Wahrnehmungspsychologie verwendet werden, die Emotionen mit verschiedenen Farbkombinationen, Kontrasten und Sättigungen in Zusammenhang bringen (Itten 1960, verwendet im Picasso System)

4.2.2 Retrieval durch Texturähnlichkeit

Texturen sind periodisch wiederkehrende Strukturen im Bildbereich, die durch ihre Frequenz, ihre Richtung, ihre Granularität beschrieben werden können. Geeignete Maßzahlen sind beispielsweise Statistische Größen wie die Autokorrelation oder Fraktale Dimension. Im Frequenzbereich kann z.B. das Powerspektrum verwendet werden.

4.2.3 Retrieval durch Ähnlichkeit der Form

Zwei grundverschiedene Varianten werden hier verfolgt: eine Form ist durch einen Featureverktor beschrieben und die Differenz wird durch eine geeignete Metrik ermittelt. Die Form wird in einzelne Teile aufgebrochen, die einzelnen Teile beschrieben und mit den Bezeihungen zwischen den Teilen im Vektor abgelegt. Andererseits können Formen unterschieden werden durch das Ausmaß an Aufwand den man betreiben muß um eine Form in eine andere überzuführen. Hier gibt es keine Möglichkeit zum Indexing.

Form durch features

Hier gibt es zwei Möglichkeiten:

- Parametrische interne Methoden: die Region selbst wird beschrieben (z.B. durch einfache geometrische Attribute wie Fläche, kleinste/größte(r) Kreis/Ellipse/Rechteck das eingeschrieben bzw. umschrieben werden kann, oder durch Zusammensetzung aus kleineren einfachen Einheiten. Digitale Momente sind eine fortgeschrittenere Möglichkeit.)
- Parametrische externe Methoden: die Regiongrenze wird als Kurve beschrieben (z.B. durch pixelbasierte Methoden - nur Verwendung von signifikanten Pixeln oder chain encoding, wo ein Gitter über die Kurve gelegt wird und die Gitterpunkte nahe an der Kurve mit Richtung zum Nächsten werden verwendet - oder Token basierten Methoden, wo die Kurve durch einfache Teile wie Polygone approximiert wird.

Form durch Transformation

Die bekannteste Variante sind Snakes. Hier wird eine Form durch Transformationen auf eine Zielform gebracht. Die Verformung wird durch eine diskrete Menge von Parametern kontrolliert. Das Ausmaß der Verformung gibt die Ähnlichkeit der beiden Formen an.

4.2.4 Retrieval durch räumliche Anordnung

Wieder gibt es zwei grundlegend verschiedene Möglichkeiten:

- Repräsentationen die örtliche Beziehungen und visuelle Information nicht trennen - objekt-basierte Strukturen. Algorithmen errechnen örtliche Beziehungen über die Untersuchung von Objektkoordinaten. Zu diesen Methoden gehören Gitter, Quadtrees, R-Trees, ...
- Repräsentationen die nur die Beziehung zwischen den visuellen Objekten darstellen, nicht aber deren visuellen Inhalt - relationale Strukturen. Die bekannteste Methode sind 2DStrings: Symbolische Bilder werden in eine eindimensionale Darstellung übergeführt, durch Projektion auf die beiden Koordinaten Achsen. Die relative Position wird durch ¡ und = ausgedrückt.

Wichtige Fragestellungen bei diesen Methoden sind, wie man Entfernungen zwischen Punkten, Objekten, Kurven, zwei Mengen von Punkten, u.s.w. mißt.

4.3 Video Shot Detection

Um Methoden des VIR auf Videos anwenden zu können, müssen aus Videos erst die sog. Keyframes extrahiert werden. Dies sind für repräsentative Frames für einzelne Shots, die meist als mittlere Frames oder erste/letzte definiert werden. Eine weitere Methode ist das sog. Mosaicing, bei dem aus allen Frames eines Shots ein einziges Bild zusammengesetzt wird (das geschieht durch Matching der unterschiedlichen Frames gegeneinander um die optimale Position zu ermitteln, aber auch affine Transformationen können notwendig werden).

Jedenfalls werden für alle erwähnten Verfahren Informationen über die Shots bzw. die Shotgrenzen im Video benötigt - ein Shot ist die kleinste semantisch zusammengehörende Menge von Frames die von einer Kamera von einer Szene aufgenommen wurden. Shots zeichnen sich durch den gleichen Aufnahmeort, gleiche Objektbewegungen und Kamerabewegungen, usw. aus. Meistens ist die Menge an Shots zu groß um für jeden shot einen Keyframe abspeichern zu können - es gibt daher Methoden um mehrere shots sinnvoll zu Szenen zusammenzufassen (z.B. wäre ein Interview eine Szene, die Beiträge der einzelnen Interviewpartner die von verschiedenen Kameras aufgezeichnet werden sind aber einzelne shots - automatisiert wird einfach nach ähnlichen Keyframes gesucht).

Im Bereich der Video Shot Detection (oder auch Scene Cut Detection) gibt es zwei grundlegend verschiedene Aufgabenstellungen:

- Scharfe Shot Übergänge: Diese werden erzeugt indem einfach zwei unterschiedliche Shots ohne Anwendung von Videoverarbeitenden Methoden aneinandergereiht werden. Solche Cuts sind einfacher zu erkennen.
- Fließende Übergänge: sog. Fades und Dissolves. Ein Fade ist das sukzessive Abdunkeln einer Folge von Frames bis ein gänzlich schwarzer Frame übrigbleibt (fade-out) - umgekehrt ist der fade-in der Übergang vom schwarzen Frame zu einem normal hellen Bild. Ein Dissolve ist eine Übereinanderlegung eines fade-out und eines fade-in. Das klassische Verfahren zur Erkennung solcher unscharfen Übergänge ist das “twin thresholding” Verfahren. Dabei wird mit zwei Schwellwerten gearbeitet: einer (der

höhere) ist für das Erkennen von scharfen Übergangen, der niederere für fließende. Wird der größere Schwellwert mit einem Frame-Ähnlickeitsmaß (siehe im Folgenden) überschritten, gilt ein Cut als erkannt. Wird nur der kleinere Schwellwert überschritten, wird der entsprechende Frame als Kandidat für den Beginn eines fließenden Übergangs identifiziert. Im Anschluss werden die Framedifferenzen akkumuliert, wenn das Akkumulat den größeren Schwellwert überschreitet und einzelne Framedifferenzen wieder unter den kleineren fallen, wird der fließende Übergang als beendet betrachtet.

Modelliert werden Fades und Dissolves durch einfache Skalierung im Farb/Grauwertbereich. Ist $G(x, y, t)$ ein Grauwertvideo und l_i die Länge des fließenden Übergangs, so sieht die Modellierung aus wie folgt:

$$Fade - out(x, y, t) = G(x, y, t)(\frac{l_1 - t}{l_1}) + 0$$

$$Fade - in(x, y, t) = 0 + G(x, y, t)(\frac{t}{l_2})$$

Shot Detection Methoden lassen sich in unterschiedliche Typen klassifizieren:

1. Algorithmen im Bildberich

   - Verwendung von globalen Features zur Ähnlichkeitsbestimmung
     - Paarweiser Pixelvergleich: die Anzahl der sich veränderten Pixel von einem Frame zum Nächsten wird gezählt, wird ein Schwellwert überschritten, gilt ein Cut als erkannt.
     - Mittelwertsvergleich: ist S_t der Mittelwert der Grauwerte des Frames t, so ist das Maß durchschnittliche Intensitätsdifferenz definiert als:

       $$D = |\frac{S_t - S_{t+1}}{S_{t-1} - S_t}|$$

     - Likelihood Koeffizient: sei $m(f_t, i)$ und $\sigma(f_t, i)$ Mittelwert und Varianz des i-ten Blocks des t-ten Frames, so ist der Likelihood Koeffizient definiert wie folgt:

       $$D(i) = [\frac{\sigma(f_t, i) + \sigma(f_{t+1}, i)}{2} + (\frac{m(f_t, i) - m(f_{t+1}, i)}{2})^2]^2$$

       Ein Cut wird detected wenn die Mehrheit der Blöcke höhere Werte für $D(i)$ liefert als ein best. Schwellwert.
     - Histogrammvergleichsmethoden: Binweise Differenzbildung, Histogramm Intersection, Chi^2 Test Histogramm Differenz ($\frac{1}{N^2}\sum_i \frac{(h_t(i) - h_{t+1}(i))^2}{h_{t+1}(i)}$).
   - Verwendung von lokalen Features: der bekannteste Ansatz verwendet Kantenbilder - die Grundannahme ist, dass bei scene cuts neue Kanten auftauchen die weit entfernt sind von Kanten im vorigen Frame, alte Kanten verschwinden die weit weg von neuen Kanten sind. Cuts werde erkannt indem die Anzahl von *entering* und *exiting* Kantenpixel in zwei aufeinanderfolgenden Frames gezählt werden. Relativ aufwendiges Verfahren, da Kantenerkennung plus die Klassifikation ob Kantenpixel in aufeinanderfolgenden Frames nah sind oder nicht durchgeführt werden müssen.

- Verwendung von "Bewegungsinformation": Meist werden hier Blockmatchingverfahren (wegen der Geschwindigkeit) eingesetzt. Beispielsweise kann der Durchschnitt der Blockmatchingfehler oder die größten Blockmatchingfehler zwischen zwei Frames als Maß verwendet werden. Wechsel von globalen Kamerabewegungen (Zoom, Pan, ...) können auch als Indikatoren für Cuts verwendet werden.

2. Algorithmen für komprimiertes Video (MPEG): in Analogie zu den vorangegangenen Methoden werden im komprimierten Bitstream zwei Hauptfeatures verwendet: wichtige Koeffizienten und Bewegungsinformation. Meist werden DC (DC-frame differences) oder DC und große AC Koeffizienten in ausgewählten Blöcken verwendet - hier können dann wieder Histogrammmethoden oder statistische Tests eingesetzt werden um Ähnlichkeit zu messen. Bewegungsinformation kann beispielsweise auf folgende Arten verwendet werden:

   - Messen der Energie in motion compensated error frames (P oder B Frames)
   - B-Frames: dominiert deutlich eine Richtung in der Prediction, kann in der anderen ein Cut sein
   - P-Frames: ist die Anzahl der nach I-Frame Methode komprimierten Makroblöcke hoch, kann ein Cut vorliegen.